UNIVERSIDADE DA CORUÑA

A Coruña

Análisis de los métodos de construcción de rutas en los sistemas de planificación para el problema del VRPTW.

Tesis elaborada en cumplimiento
de los requerimientos para acceder al título de
Doctor en CC. Económicas y Empresariales

por

Eduardo Guillén Solórzano

2003

Dirigida por

Dr. D. Alejandro García del Valle

ÍNDICE DE CONTENIDOS

Capítulo 1 Revisión del Problema

Capítulo 2 Estado de la Cuestión

Capítulo 3 Desarrollo del Modelo

Capítulo 4 Experimentación y Resultados

Capítulo 5 Conclusiones y Futuras Líneas de Investigación

LISTA DE FIGURAS

LISTA DE TABLAS

AGRADECIMIENTOS

En este punto, y antes de comenzar con el contenido de este trabajo, quisiera mostrar mi agradecimiento a todas aquellas personas sin las cuáles este proyecto no hubiera sido posible. En primer lugar, a mi director y tutor durante la presente investigación, el profesor Dr. D. Alejandro García del Valle, por su constante apoyo y motivación para el desarrollo de este trabajo, así como sus valiosos comentarios y correcciones que permitieron que este proyecto llegase a buen puerto. A mi familia, y en especial a mi esposa, por los constantes ánimos prestados durante todo este tiempo de esfuerzos y trabajo.

Igualmente me gustaría aprovechar la ocasión para agradecer los apoyos prestados para la realización de esta Tesis. A la Escuela Universitaria de Diseño Industrial de la Universidad de La Coruña, y en especial a su director, el profesor D. José Ramón Méndez Salgueiro, por haber facilitado la utilización de los medios informáticos de la Escuela. Al Laboratorio de Lenguajes de Programación de la Universidad de La Coruña, *RNASA*, y en especial a José Santiago Cernadas Vilas, por su inestimable colaboración en el desarrollo del código de programación, y a todos mis compañeros. También quisiera indicar y agradecer muy especialmente la colaboración prestada por el equipo de investigación merecedor de la ayuda del Ministerio de Fomento que lleva por título "*Optimización de rutas de transporte para el Reparto de Mercancías*", y que permitió la financiación parcial de algunas de las actividades necesarias para abordar este proyecto, compuesto por Dr. D. Alejandro García del Valle, Dr. D. Javier Faulín Fajardo y D. Arturo Nieto de Almeida.

Gracias a todos por vuestras colaboraciones y aportaciones al presente trabajo.

RESUMEN DE LA TESIS

En este trabajo se aborda la problemática de la planificación de rutas de transporte para el reparto de mercancías tanto en empresas industriales como en empresas de transportes de carga fragmentada. A pesar de que se trata de una problemática que afecta por igual a grandes y a pequeñas empresas, se ha centrado la atención en las PYMES, donde la problemática relativa a la distribución obedece más a problemas operativos, que no a cuestiones estratégicas.

La justificación de la importancia que tienen y sobre todo que en un futuro tendrán los sistemas de planificación de rutas es sobradamente conocida, dado que deriva de causas que son en gran parte de sentido común: la escalada de precios de los combustibles, la creciente competencia por grandes multinacionales, las oportunidades que se presentan a raíz del comercio electrónico, o el fenómeno de la globalización, provocan que cada vez sea mayor la proporción de los costes de transporte en el cómputo del coste total unitario de los productos. Por esto la búsqueda de sistemas de planificación más racionales, eficaces y eficientes, puede aportar enormes ventajas competitivas a las empresas.

En los últimos años han proliferado los trabajos y las actuaciones conducentes al desarrollo de Sistemas de Información Logística muy potentes, encaminados a la resolución de los problemas relativos a la planificación y programación de rutas de transporte, motivado en gran parte por los avances tecnológicos conseguidos en múltiples campos, y que se refunden en estos sistemas avanzados de planificación. Entre estos cabe destacar la importancia actual de los Sistemas de Información Geográfica, los sistemas de geoposicionamiento y geolocalización, los sistemas de seguimiento de flotas de vehículos, así como los avances en potencia de cálculo de los ordenadores, y en general el abaratamiento de las tecnologías. En este sentido, los Sistemas de Información Logística se sirven de todos estos avances para dar lugar a potentes herramientas de optimización que buscan en todo momento agilizar la tarea de planificación y programación de servicios, así como el objetivo último de la reducción de costes, que por término medio se consigue en cifras que oscilan entre el 15 y el 30% de los costes en los que se incurriría con una planificación manual.

Este problema de la planificación de rutas de transporte se conoce en la literatura como VRP o *Vehicle Routing Problem*, que deriva de la problemática del TSP, o *Traveling Salesman Problem*, en la que un vehículo ha de encontrar la ruta mínima entre un conjunto de puntos geográficamente dispersos. En el caso del VRP, se generaliza el problema del TSP, y se contemplan un número *NV* de vehículos, con unas limitaciones de tiempos, o de cargas, de manera que es necesaria la utilización de esta flota de vehículos para visitar todos los puntos del problema. Desde mediados del siglo XX, esta problemática fue abordada por la línea de Investigación Operativa, donde se enmarcan la mayor parte de los trabajos existentes, y cuya evolución ha sido notable sobre todo en los últimos 10 años. Merece la pena destacar una fecha, 1987, que es cuando Marius M. Solomon recoge toda la información relativa a los sistemas existentes para la planificación de rutas de transporte en el marco del VRP, y los adapta al problema del

VRP con *Time Windows* (VRPTW), o VRP con Ventanas de Tiempo, con el ánimo de acercar todavía más la línea de investigación existente a la problemática empresarial. La peculiaridad del VRPTW deriva en la consideración de intervalos de apertura para cada uno de los clientes, así como los plazos de apertura del depósito central en el que se encuentran los vehículos. De esta manera se aporta un mayor realismo al problema general del VRP. Para ello Solomon (1983) recoge en su Tesis Doctoral esta problemática, publicando en 1987 uno de los artículos más recurrentes en todos los trabajos en el que resume los resultados de aplicación de diferentes métodos de construcción de rutas al problema del VRPTW. Igualmente desarrolla una batería de ejemplos sobre los que valida estos desarrollos, y que servirían de base para la inmensa mayoría de los autores con la finalidad de normalizar los resultados de los diferentes métodos aplicados.

Es por lo tanto en la última década del siglo XX cuando se obtienen los mayores avances en los sistemas de resolución del VRPTW, sobre todo a través de la implementación de técnicas de aproximación matemática de última generación, como los algoritmos genéticos, la búsqueda tabú, o el recocido simulado. Sin embargo, no se desarrolló tanto la rama dedicada a los métodos de construcción de rutas, motivado en parte por la superioridad de las técnicas de aproximación anteriormente mencionadas. Es destacable que todas ellas dependen de los métodos de construcción, y se sirven de ellos para construir una primera solución sobre la que implementar las técnicas de postoptimización. Por ello en este trabajo se ha profundizado en estas técnicas de construcción, para comprender los principios en los que se apoyan, y aprender de los mismos para mejorar los métodos existentes.

En este marco general se integra el presente trabajo que se ha estructurado en cinco capítulos, y otros tantos anexos. En primer lugar se presenta la investigación realizada en empresas de transporte y en empresas de distribución en las que se ha analizado de forma somera las reglas que guían la planificación de las rutas de transporte. De forma muy genérica se comenta la problemática empresarial en cuanto al desarrollo de estas rutas. Igualmente en este primer capítulo se recoge un análisis de las diferentes soluciones informáticas existentes para resolver el problema de la planificación, destacando que la mayor parte de ellas provienen de Estados Unidos, y que la aplicabilidad de las mismas al caso español está condicionada a la existencia de información geográfica sobre el territorio, y que hasta la fecha ha supuesto un problema a la hora de implantar estos sistemas en España.

En el capítulo 2, se presenta una recopilación de los diferentes trabajos relativos a los sistemas de planificación de rutas de transporte, y que se han enmarcado en los problemas del TSP, VRP, y VRPTW. Si bien es este último el que se estudia con detenimiento en este trabajo, ha sido necesario recopilar información sobre las tres problemáticas dada la elevada interdependencia existente entre las tres. Por ello se analizan los trabajos más destacables en las tres áreas, prestando especial atención a los mecanismos de generación de rutas. Por último en este capítulo también se comentan algunos ejemplos de aplicación

de estas herramientas a casos reales, destacando mayormente por resultar trabajos ad hoc, y no soluciones genéricas, como ocurría en el capítulo 1.

En tercer lugar, se aborda el desarrollo de un método propio para la resolución del problema del VRPTW. Este método se enmarca dentro de los algoritmos de construcción de rutas de transporte, y por ello incurre en todas las ventajas de los mismos, particularmente su enorme eficiencia motivada por la velocidad de cálculo de soluciones, su determinismo, en el cómputo de las mismas, así como su enorme flexibilidad dado que se trata de un algoritmo parametrizado. Sin embargo también peca de los inconvenientes de estos métodos, y que se resumen en la mediocridad de las soluciones obtenidas. La finalidad de este desarrollo no es otra que aprender de los métodos de construcción de rutas, así como de desarrollar un método propio para resolver este problema con la finalidad de proponer futuros desarrollos, y buscar posibles aplicaciones de este método a problemas empresariales.

En el capítulo 4 se presentan los resultados conseguidos por el algoritmo desarrollado sobre la batería de problemas tipo desarrollada por Solomon (1987). La presentación de estos resultados se realiza de forma individual para cada uno de los 56 problemas, para los que se aporta la solución de las rutas en cada caso. Igualmente se comenta las peculiaridades de cada uno de los 6 tipos de problemas diferentes, destacando los grupos *R2* y *RC2*, donde el algoritmo consigue superar los resultados aportados por otros métodos de construcción de rutas, e incluso de algún método de postoptimización. Es precisamente los problemas que más se asemejan a la problemática empresarial relativa a las PYMES del sector de transportes, y que se corresponden con la visita de clientes geográficamente dispersos, no excesivamente distantes del depósito central, de manera que las rutas resultantes integrarán un número de visitas elevado (entre 15 y 30 visitas por vehículo), correspondiéndose con un horizonte de planificación amplio. Generalmente se trata del "*transporte de corta distancia*", con una elevada incertidumbre en el número y ubicación de los servicios. Igualmente en este capítulo se compara la bondad de ajuste de este método con otros métodos desarrollados en la literatura, así como un análisis de sensibilidad de los parámetros utilizados por el mismo. Por último se detectan y analizan posibles áreas de mejora para el algoritmo con la finalidad de incrementar la eficacia del mismo y buscar una máxima aplicabilidad a situaciones reales.

Por último en el capítulo 5 se comentan las conclusiones más relevantes del trabajo realizado, así como se analizan posibles desarrollos del algoritmo. Para ello se presta especial atención a la integración de las últimas tecnologías relativas a los Sistemas de Información Geográfica, así como las comunicaciones y sistemas de localización vía satélite.

De esta manera se cierra la investigación realizada, con la conclusión final de que se trata de un área en la que actualmente se presentan multitud de oportunidades de negocio tanto para las empresas de transporte, como para las empresas de desarrollos informáticos, y que sin duda requerirán de las investigaciones científicas previas sobre el VRPTW aquí recogidas, y de las que este trabajo pretende ser un ejemplo más.

Capítulo 1

Revisión del problema

La intención del presente capítulo es justificar la importancia actual, y sobre todo la importancia que tendrá en los próximos años, el desarrollo de los sistemas de planificación de rutas de transporte, la racionalización de las actividades de distribución y la optimización de rutas de reparto. Para ello se ha dividido en tres epígrafes. En primer lugar se comentan a grandes rasgos cuáles son los fenómenos que afectan al sector de la distribución y del transporte. En segundo lugar y adoptando un enfoque más microscópico, se centra la atención en el caso de la empresa de distribución o de transportes y, en general, de cualquier empresa que desarrolle actividades de logística externa. En este apartado se discute de forma más detallada los problemas operativos de esta clase de empresas y en mayor profundidad la manera de programar las rutas de reparto. En último lugar se exponen las diferentes soluciones existentes en la actualidad para afrontar este tipo de problemas logísticos y de manera concreta, se centra el estudio en las soluciones informáticas, tanto para la optimización de rutas como para la visualización de las mismas empleando los conocidos Sistemas de Información Geográfica (GIS).

1 Problemática general del sector

En este apartado se comentan de forma general los problemas y dificultades que afronta el sector de la distribución y el transporte. Como primer comentario cabe destacar la gran importancia de este sector en todas las economías desarrolladas como uno de los pilares básicos del desarrollo sostenible ante la creciente internacionalización de los mercados. Además es necesario destacar que el coste derivado del transporte de cualquier tipo de producto tiende a incrementarse debido a una serie de circunstancias que se discuten en las siguientes líneas. De ahí surge la gran necesidad de abaratar los costes de transporte a partir de una distribución más racional de las mercancías. En concreto, se plantean los fenómenos de la globalización, el desarrollo del comercio electrónico, el encarecimiento de los carburantes, la mejora de infraestructuras y la creciente concentración del sector.

1.1 Globalización e internacionalización

Es conocido el fenómeno de la globalización e internacionalización de los mercados, algo que cada vez está cobrando más importancia a la hora de competir en un sector o industria determinado. Las economías nacionales, o incluso regionales, se ven afectadas en gran medida por la apertura de estos mercados a nuevos competidores. Sin embargo, este fenómeno tiene una doble interpretación: por un lado la situación de las empresas locales de un determinado país se topan con nuevos competidores, en muchos casos mejor preparados que las primeras, pero por otro lado, estas empresas locales se encuentran con una puerta abierta hacia el exterior que les permite acudir a mercados extranjeros.

Este fenómeno, que provoca tanto amenazas como nuevas oportunidades, ha de ser aprovechado por todos los sectores, lo cual sólo es posible si existe una importante red de distribución propia, o bien el suficiente número de empresas de transporte externas que garanticen un servicio de calidad en la distribución de los productos.

Este hecho justifica la importancia del desarrollo de redes de transporte más eficientes, algo que destaca más si tenemos en cuenta que el porcentaje de costes de distribución en los productos se ha ido incrementando desde hace algunos años, bien porque las materias primas se importan desde países extranjeros, bien porque se fabrica en países lejanos donde las condiciones de mano de obra son más favorables para la empresa, o bien porque se está sirviendo a mercados lejanos.

En el caso concreto del sector de transportes, en los últimos años se ha detectado un creciente desembarco de empresas extranjeras en el mercado nacional. Entre ellas destacan algunos nombres conocidos en el ámbito internacional como por ejemplo, *UPS*, *FEDEX*, o *Danzas*. Esta incursión en mercados foráneos generalmente obedece a la necesidad de proporcionar un servicio integral a sus clientes y que en muchos casos pasa por realizar envíos internacionales. Del mismo modo, las empresas nacionales también tienden a acudir a mercados externos. Empresas representativas de este fenómeno pueden ser *Seur*, *Azcar*, *Ochoa*, *Gefco*, *Teisa*, *Spain Tir*, o *Transerra*, aunque si bien es cierto, la

mayor parte de las operaciones internacionales se realizan a nivel más europeo que global.

En un reciente artículo de la revista "*Logística, Paquetería y Almacenaje, Nov 2000*"[1] se publicaba un informe sobre la situación del transporte de mercancías por carretera en el mercado europeo. En este sentido su subdirector, D. Joaquín del Palacio, ponía de manifiesto el proceso de integración de las redes de carreteras de todos los mercados nacionales de Europa, en una única red europea de carreteras que, aunque siempre han estado geográficamente unidas, es en estos momentos cuando los operadores de transporte de mercancías contemplan este mercado como un mercado único. Si bien existen todavía barreras a este proceso, como puede ser el idioma o el desconocimiento de las carreteras locales, de manera que algunas empresas para realizar este tipo de actividades transeuropeas se sirven de transportistas locales para realizar los repartos de la última fase de la cadena. Sin embargo, esto está cambiando en los últimos años debido a los factores apuntados anteriormente, y principalmente al desarrollo de las tecnologías de la información, como se verá en la última parte de este capítulo.

1.2 Desarrollo del comercio electrónico

Como se comentaba al principio, es necesario tener en cuenta el desarrollo de las nuevas tecnologías y costumbres sociales. En especial cabe destacar la creciente evolución del comercio electrónico. En un futuro próximo se prevé un fuerte crecimiento del comercio electrónico a través de *Internet*. El desarrollo de las nuevas tecnologías facilita cada vez más la compra desde casa, de forma que el establecimiento tradicional se encontrará con una fuerte competencia por parte del comercio electrónico.

A pesar de estos augurios, ya se han detectado algunos frenos a esta visión que se realizaba a finales del siglo XX. En este sentido, el profesor Randolph Hall (Catedrático de Ingeniería industrial y de sistemas de la Universidad de California del Sur, y uno de los colaboradores de la encuesta sobre software aplicado a la planificación de rutas de transporte) apuntaba en febrero del año 2002 que se estaba produciendo una recesión en el comercio electrónico en el mercado estadounidense, lo que presionaba a las empresas de transporte a replantearse las estrategias destinadas a cubrir las necesidades de estos mercados. Todavía existe una cierta desconfianza a comprar a través de *Internet*, de manera que no se está alcanzando las previsiones iniciales. Sin embargo, la mayor parte de las empresas mantienen estas líneas destinadas hacia los clientes finales o consumidores.

Parece inevitable que, tanto a nivel internacional como a niveles más locales, el desarrollo de las redes de distribución sea cada vez más una fuente de ventajas competitivas para la empresa. Por ello el desarrollo efectivo y eficiente de este tipo de redes de distribución se convertirán en un factor determinante, teniendo en cuenta que el coste de transporte cada vez será más elevado y por ello pequeñas reducciones en los costes de este componente implicarán grandes reducciones en el coste final del producto.

[1] Véase http://www.logisticaytransporte.net/

Dentro del sector de transportes se ha planteado en numerosas ocasiones la necesidad de desarrollar estas redes de ámbito nacional o regional que permitan el suministro diario de los productos consumidos a través del comercio electrónico. En este sentido caben dos posibles alternativas:

- Desarrollar una nueva red de distribución total en el territorio español, que sería utilizada única y exclusivamente para realizar estas entregas. Dado que la situación del desarrollo del comercio electrónico como forma de consumo todavía no está lo suficientemente instaurada en las costumbres sociales, esta alternativa se ha descartado.
- Utilizar las redes de distribución existentes para el reparto de estos productos, ya sea a través de la distribución de los mismos por empresas de transporte existentes, o bien a través de las redes más tradicionales como es el caso de *Correos y Telégrafos*.

El comercio electrónico también está provocando fuertes cambios en la manera de vender los servicios de transporte, tanto a las empresas (línea *B2B, Business to Business*), como en los servicios destinados a los clientes finales (línea *B2C, Business to Customer*). En este sentido, las aplicaciones del comercio electrónico a las empresas de transporte también están dando sus frutos. La necesidad fundamental es la del establecimiento de sistemas de seguimiento de los envíos para los clientes. De esta manera los clientes, a través del código de envío son capaces de realizar un seguimiento on-line a través de la página de *Internet* del proveedor, conociendo en todo momento la situación y localización del envío, así como su próxima fecha de entrega. Estos servicios de seguimiento o *tracking*, ya están disponibles en la mayor parte de las empresas de transporte multinacionales. También se trata de una exigencia por parte de las normativas de calidad cada vez más demandadas por los clientes de las empresas de transporte para tener una garantía de los servicios de las mismas.

El hecho de implementar este tipo de sistemas de seguimiento de envíos, deriva de los propios sistemas de seguimiento que las empresas implantan para controlar la situación y estatus de cada uno de los vehículos de los que dispone, y solamente a través de este tipo de sistemas es posible proporcionar a los clientes información en tiempo real de la situación de sus envíos. También se contribuye a proporcionar una mayor seguridad a los posibles clientes de empresas industriales que adquieran sus productos a través de *Internet*, ya que psicológicamente aporta mayor seguridad y confianza, una vez que se puede realizar el seguimiento de cualquier pedido en tiempo real.

Las aplicaciones de seguimiento de flotas de vehículos son hoy día una realidad en muchas empresas, ya que aportan innumerables ventajas a la planificación de las rutas de transporte. En los últimos años se han desarrollado un sinfín de tecnologías aplicables a esta tarea, tanto fundamentadas en los conocidos GPS (*Global Positioning Systems*), como en las fundamentadas en redes inalámbricas de transmisión de datos como es la conocida GSM (*Global System for Mobile Telecommunications*), o la tan ansiada esperada tecnología UMTS (*Universal Mobile Telecommunications Services*), que hace

aparición a principios del siglo XXI. Algunas de estas ventajas se resumen a continuación:

- Conocimiento de la situación de cada uno de los vehículos
- Posibilita la planificación dinámica de las rutas
- Permite la planificación de contingencias
- Permite una comunicación constante entre el vehículo y la central
- Control de aquellos parámetros del vehículo que tengan interés para el planificador: velocidad, régimen de revoluciones, consumos medios, etc.
- Permite el seguimiento de los envíos a través de *Internet* para los clientes.

Una explicación más detallada de la implantación de estos sistemas se desarrolla en el capítulo 5 sobre Conclusiones y Futuras Líneas de Investigación. Generalmente la implantación de los sistemas de seguimiento de flotas de vehículos son necesidades requeridas por los clientes de las empresas de transporte. En este sentido se presenta una oportunidad no solamente de implantar estos sistemas para mejorar el servicio ofertado, sino también de implementar sistemas de planificación y optimización de las rutas de transporte, con un coste adicional muy reducido. Es por este motivo que la implantación tanto de los sistemas de seguimiento de flotas, como de los propios sistemas de planificación de rutas, y optimización, sean simultáneos de manera que se integran en los denominados sistemas SIL, o Sistemas de Información Logística en empresas de transporte de mercancías por carretera. Una visión más completa de este tipo de software se realiza en la última sección de este capítulo.

1.3 Incremento de los precios de los carburantes.

A finales del siglo XX se ha producido una nueva escalada de precios de los carburantes, aunque no tan drástica como la sufrida en la crisis del petróleo de 1973, que provocó una gran recesión en las economías mundiales. En este caso, la escalada de precios supuso para los carburantes un incremento de entre el 15 y el 20% en un período reducido de tiempo de entorno a unos 6 meses, a finales del año 2000 y comienzos del 2001.

El efecto de esta subida de precios de los carburantes tuvo sus mayores consecuencias en el sector de transporte dado que muchas de las empresas tenían sus tarifas fijadas con sus clientes a modo de contrato de suministro de manera que la subida de los carburantes provocó graves pérdidas en el sector, únicamente paliadas en parte por la autorización general para que los transportistas subiesen sus tarifas un 10% de los precios fijados.

Figura 1.1 Evolución de la cotización del barril de crudo.
Diario Expansión www.expansion.es

Sin embargo los efectos de la subida de carburantes no se estancaron en el sector de transporte, sino que a través del efecto dominó, se reflejaron en la consiguiente subida de los productos que las mismas suministraban de forma que se produjo un incremento general de los precios, tanto de los productos industriales como los de consumo. Estas subidas hicieron repuntar todos los índices de inflación, tanto los industriales como el Índice de Precios al Consumo, cuya evolución se expone en la figura 1.2.

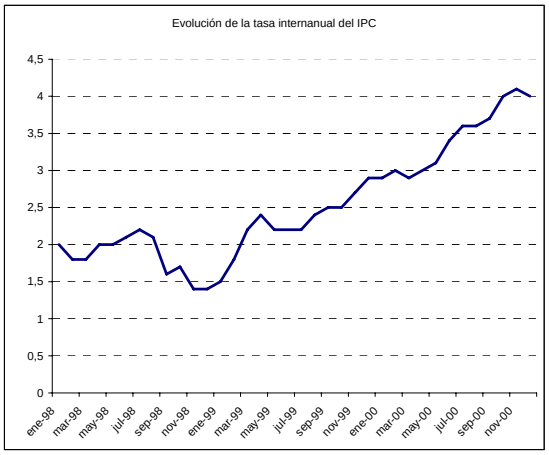


Figura 1.2 Evolución de la tasa interanual del IPC.

Además de las subidas de todos los productos en general también provocó efectos en las movilizaciones de sindicatos y empresas de transportes, que derivaron en importantes huelgas que paralizaron al país durante algunos días dejando sin suministro a muchas empresas, que se vieron obligadas a parar sus procesos productivos, generalmente basados en el suministro Just in Time por parte de sus proveedores. (Sirva de ejemplo el caso de *Citröen* en Vigo, que tuvo que fletar varios aviones Antonov para garantizar el suministro de motores a su planta en la Zona Franca).

Las causas de estas oscilaciones de los precios en los carburantes son conocidas y generalmente se resumen en dos:

- La dependencia de suministro de los países no productores de petróleo del oligopolio de países productores, organizados entorno a la OPEP.
- El hecho de que las transacciones realizadas por empresas y países para comprar crudo se realicen en dólares, de manera que el precio global del producto está enormemente afectado por el riesgo de tipo de cambio.

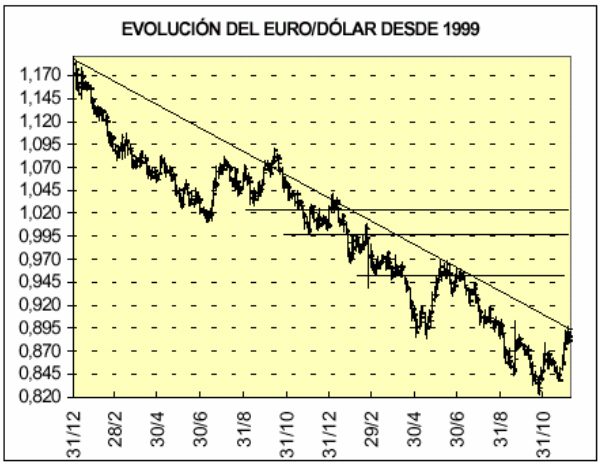


Figura 1.3 El tipo de cambio €/$. Expectativas a medio plazo. 2000
Dirección financiera Caja Madrid. Área de estudios

Por ello la tendencia de los precios de los carburantes es eminentemente alcista, lo cuál a largo plazo se acentuará si tenemos en cuenta que los recursos son limitados.

1.4 Mejoras de las infraestructuras

Las mejoras en las infraestructuras y, más en concreto, en las vías de comunicación, provocan que empresas que inicialmente no operaban en determinadas zonas, se planteen ampliar su área de influencia. Este fenómeno también viene motivado por la caída de las barreras nacionales que si bien no impedían el transporte internacional, sí lo dificultaban enormemente. Esta apertura de nuevas oportunidades generalmente se ven aprovechadas en gran medida por las empresas mejor posicionadas en el sector y con grandes niveles de recursos. Es destacable el incremento que se está produciendo actualmente en el volumen de tráfico de mercancías por carretera, en contra de la disminución de otro tipo de medios, como el transporte por ferrocarril.

Las mejoras en las infraestructuras también hacen referencia a las nuevas tecnologías. El papel de las nuevas tecnologías en el sector de transporte es fundamental. Desde la recepción de los paquetes en el origen, hasta la entrega de los mismos en su destino, se produce una concatenación de actividades relacionadas con las nuevas tecnologías y las aplicaciones informáticas que permiten que todo el proceso se realice de una manera

mucho más ágil y en condiciones de mayor seguridad de lo que se realizaba antes. Por ejemplo, la aplicación de la codificación a los paquetes, bien con códigos de barras, o bien con etiquetas de radiofrecuencia, permite una distribución física en el almacén hacia las diferentes rutas, de manera completamente automatizada.

Todo ello ha permitido incrementar los niveles de satisfacción de los clientes de las empresas de transporte y consecuentemente sus exigencias. Es destacable la importancia que se otorga por parte del cliente que los servicios tengan lugar en un día determinado y en un intervalo de tiempo concreto, algo que en la literatura se conoce como *ventanas de tiempo*. Este problema supone una mayor complejidad para la empresa de transporte, pero que ha de afrontar necesariamente dado que supone otra exigencia del mercado. Es por ello que una mayor racionalización del proceso se hace necesaria para implicar a las empresas en esta mejora continua, que persigue en todo momento la garantía de que las necesidades y exigencias del cliente se verán cumplidas con toda fiabilidad.

1.5 Creciente concentración del sector

En los últimos años se está produciendo una gran concentración en el sector de las empresas de transporte. Esta concentración deriva, como se comentaba al principio, de la desaparición de las barreras nacionales y la internacionalización de las economías, que provoca que las empresas tengan que enfrentarse a grandes multinacionales provenientes de otros países. Por ello, la concentración es la salida inmediata para paliar estas amenazas. Esta concentración está teniendo lugar tanto en el transporte de mercancías como de pasajeros.

Las ventajas de estas fusiones de empresas son evidentes, ya que en muchos casos se reducen recursos duplicados (talleres, personal de administración, rutas, almacenes...), se aprovechan sinergias y además se goza de las ventajas derivadas de las economías de escala, tanto para adquirir infraestructuras (por ejemplo, el caso de adquisición de vehículos) como para obtener ventajas en los costes variables, generalmente en el suministro de carburantes.

Esta concentración está provocando la creación de fuertes grupos empresariales, que provocan una situación de debilidad de las PYMES del sector. Mientras que las grandes compañías, dotadas de grandes volúmenes de recursos, se centran en las largas rutas y en el transporte internacional, las pequeñas empresas, quedan relegadas a realizar los repartos de ámbito más local y en muchos casos en zonas poco rentables para la gran empresa. Este desplazamiento de las PYMES implica, en muchos casos, también asumir el papel de subcontratista de una gran empresa para establecer la delegación de la misma en una zona remota.

En este sentido y puede que sea el futuro del sector, nos podemos encontrar con grandes empresas, que generalmente gozan de reconocido prestigio en el mercado, cubriendo sus actividades críticas y copando la mayor parte del mercado, mientras que detrás de estas grandes marcas se hallan pequeñas empresas que sobreviven realizando los transportes físicos propiamente dichos y que son las últimas responsables de la entrega del producto,

en un momento determinado y con unas condiciones también determinadas. Así el Vicepresidente Ejecutivo del grupo *Seur España*, afirmaba en el año 2001, que en un futuro próximo en algunos sectores de la actividad de transporte, como en el caso de correo postal, nos encontraríamos con situaciones oligopolísticas de mercado, donde en Europa llegarían a operar cinco o seis operadores únicos, aprovechando recursos comunes, y optimizando costes, principalmente derivados de las economías de escala y de experiencia de las empresas supervivientes.

Por ello es habitual encontrarse con situaciones en las que muchas de las pequeñas empresas de transportes de ámbito local son a su vez las representantes, o delegadas de una gran empresa de ámbito nacional, que acude a ellas para servir una zona remota, sobre todo en los últimos años, donde el sistema de franquicias de servicios también está desembarcando en el sector del transporte, de manera que es habitual que las empresas nacionales o internacionales operen en mercados más locales a través de franquicias, que disfrutan de la marca y prestigio de estas empresas, así como de la tecnología y know-how de las mismas, siendo los distribuidores de estos servicios hacia los clientes finales. (Un ejemplo de este tipo de empresas es *Seur España, S.A.*) Algunos datos avalan estas afirmaciones:

- En el informe de la *Asociación Española de Couriers Internacionales* sobre el sector en España en 1999, indicaba que el número de empresas con delegaciones propias era de 216, mientras que el número de agencias en exclusiva ascendía a 467, y por último, el número de franquicias era de 848, con un total de 3.997 vehículos de reparto funcionando diariamente en todo el territorio nacional para el reparto de paquetería.

- De estas empresas, 14 utilizaban equipos de comunicación móviles entre la flota de vehículos y los almacenes centrales, 11 utilizaban sistemas de seguimiento de envíos por códigos de barras, otras 9 sistemas de control de llamadas, y otras 7 otro tipo de sistemas de seguimiento.

- En España se realizaron en 1999 un total de 11.160.333 envíos de paquetes ascendiendo a un total de 101.011.326 Kg. de carga total.

A su vez, estas pequeñas empresas, con flota propia y recursos propios, también suelen aplicar la misma política para cubrir sus puntas de demanda, acudiendo a profesionales autónomos para servir determinadas rutas lejanas, o zonas donde generalmente hay muy pocos servicios. De manera que la figura del profesional es la encargada de aprovechar estos servicios residuales.

En este sector también se caracteriza por su gran fragmentación en cuanto al número de autónomos que operan en él. Algunas de estas PYMES están encontrando un lugar protegido de mercado a través de la especialización en el transporte de un determinado tipo de mercancías, de manera que logran su objetivo de supervivencia en estos nichos de mercado, como por ejemplo el reparto de prenda colgada, mercancías peligrosas,

productos congelados, etc, donde las empresas grandes encuentran prohibitivo entrar a operar.

Debido a esta situación, las grandes empresas están aprovechando las nuevas tecnologías para desarrollar sistemas y aplicaciones para la planificación de rutas de transporte a nivel global, mientras que las PYMES están en desventaja dado que carecen de los recursos y el saber hacer necesarios para el desarrollo de estas aplicaciones. Por ello se hace patente la necesidad de desarrollar sistemas para las PYMES en el ámbito de la planificación de rutas.

1.6 Mayor implantación de políticas de calidad total

Otra de las tendencias que afectan generalmente a todos los sectores de actividad económica y en concreto al sector de transportes, es la implantación de sistemas de calidad. En los últimos años la creciente preocupación por las empresas de obtener los certificados de calidad en sus sistemas de gestión (mayoritariamente en la norma ISO 9001). Para ello las empresas han de estandarizar todas las actividades que puedan afectar a la calidad de sus servicios. Estas estandarizaciones se desarrollan a través de la descripción detallada de todos los procesos relevantes en la empresa.

En el caso del sector de transportes, o de la distribución, es necesario definir y concretar las actividades de planificación de rutas. Para ello se plantea la necesidad de desarrollar un sistema de planificación coherente, racional y sistemático. Sin embargo las PYMES del sector generalmente recurren a la planificación intuitiva para el desarrollo de sus rutas de reparto y aunque en algunos casos sí resultan eficaces, muchas veces es posible desarrollar métodos mucho más eficientes y eficaces. La ventaja de estos métodos es que proporcionan un conocimiento explícito a la empresa y no tácito, como ocurre en la planificación intuitiva tradicional.

1.7 Otros factores de influencia

Otros fenómenos del sector, basados en los estudios realizados por el equipo de investigación de *The GreenTrip Project*[2] en el ámbito de la logística, son la tendencia hacia un creciente número de paquetes pequeños, en contra de la carga total; la introducción de estrategias *Just in Time* en un mayor número de empresas y un mayor énfasis en el tiempo de servicio, ya sea por la necesidad de servicios más rápidos como por la exigencia de cumplir determinadas ventanas de tiempo (también provocado por estrategias *Just in Time*).

Además destacan la existencia de factores externos al sector como por ejemplo una mayor preocupación por el medio ambiente y el crecimiento sostenido, que implica que los recursos se protejan más y que se reduzcan los niveles de contaminación y residuos a nivel general. También la existencia de normativas que restringen de alguna manera la actividad logística, como por ejemplo los descansos obligatorios de los conductores, la

[2] The GreenTrip Project: Pirelli (I), Tollpost-Globe (N), ILOG (F), University of Strathclyde (UK) y SINTEF (N). Email: GreenTrip@math.sintef.no, http://www.oslo.sintef.no/GreenTrip/

prohibición de circular a determinadas horas para ciertas mercancías y el incremento general del tráfico en las vías de comunicación.

Debido a esto es necesario comprender y desarrollar sistemas de gestión que nos permitan alcanzar una racionalización de la planificación de las rutas, principalmente para las pequeñas y medianas empresas que son las responsables de entregar los productos a los clientes finales, tratando en todo caso de reducir los costes de las rutas, pero manteniendo un nivel de servicio que garantice la satisfacción de las necesidades de los clientes, algo que en muchos casos es contradictorio y de ahí deriva gran parte de la problemática de este tipo de sistemas. Generalmente las oportunidades de mejora en la racionalización de los mecanismos de planificación de rutas mediante sistemas globales de coordinación, oscilan entre el 15% y el 30% reportado por algunas experiencias descritas en la literatura.

1.8 Conclusión

Con este planteamiento, el objetivo de este trabajo consiste en analizar los procesos de decisión para la planificación de las rutas de transporte, con la finalidad de elaborar un método que trate de minimizar los costes de distribución de una empresa, teniendo en cuenta un alto grado de satisfacción de las necesidades de los clientes, basadas en su mayor parte en garantizar la disponibilidad de los productos en el momento adecuado, lo que deriva del establecimiento de una serie de ventanas de tiempo por parte de los clientes, que han de ser respetadas por la empresa. Para ello se presenta el análisis realizado sobre los sistemas de planificación de rutas de transporte en dos frentes complementarios. Por un lado, el análisis de las decisiones empresariales en cuanto a la generación de las rutas de transporte, y la programación de las visitas a realizar, y en segundo lugar, los métodos de programación de visitas y de planificación de rutas que se han desarrollado en los últimos años en la línea de Investigación Operativa.

El fundamento de la investigación parte de la problemática del TSP (*Traveling Salesman Problem*, o Problema del Viajante), que si es generalizado múltiples vehículos se convierte en un VRP (*Vehicle Routing Problem*, o Problema de Rutas de Vehículos) y que unido al establecimiento de determinados plazos para la realización de los servicios a los clientes se transforma en el problema del VRP con Ventanas de Tiempo, más conocido como VRPTW, o *VRP with Time Windows.*

1 La problemática empresarial

En la sección anterior se trataba la problemática general del sector de distribución y transporte de mercancías. En este apartado la intención es reducir el enfoque para centrar la atención en la forma de planificación de las empresas con problemas logísticos (Ver tabla 1.1). Para ello observamos cuáles son los problemas típicos a la hora de establecer una red de rutas de servicio o de visitas a diferentes clientes de la empresa, intentando buscar una correlación entre cada uno de estos problemas y su nivel de decisión correspondiente.

Generalmente en cualquier empresa que se encuentre con este tipo de problemas, las decisiones relativas a los mismos se pueden asociar a los tres niveles jerárquicos clásicos. En primer lugar, desde el punto de vista de planificación estratégica, la empresa ha de decidir el diseño y configuración de la red de transportes, indicando la situación de sus almacenes, la capacidad de los mismos, los recursos asociados a cada uno de ellos y el sistema de información utilizado (denominados Sistemas de Información Logística, SIL). En el siguiente nivel y desde el punto de vista táctico, la empresa ha de determinar, en base a la demanda de sus clientes, el tamaño de la flota y la composición de la misma, los tipos de vehículos a adquirir y su variedad en la empresa, así como subcontratar los posibles excesos de demanda. Finalmente, en el nivel más operativo, la empresa ha de tomar las decisiones del día a día, la planificación de las visitas a realizar por los vehículos, los momentos de esas visitas y la asignación de esas visitas a cada uno de los vehículos.

Sectores	Actividades
Industria de automoción	Distribución de recambios a talleres
Envío de materias primas	Aceite, gasoil, cemento
Transporte de comida	Cadenas de distribución o pequeñas tiendas
Comidas a domicilio	Comida congelada
Comercio minorista	Envíos a domicilio de pedidos.
Sanidad	Medicinas a las farmacias
Prensa	Periódicos y revistas
Industria bancaria	Envíos de efectivos a sucursales y cajeros automáticos, recogidas de efectivo en tiendas y grandes almacenes.
Sector Público	Recogida de basuras, rutas de barrenderos, rutas de carteros, rutas de autobuses urbanos y de buses escolares.
Fabricación	Reorganización de movimientos de los robots de fabricación en la planta productiva, fabricación de tarjetas de circuitos integrados, cortes automáticos de prendas.
Agricultura	Recogida de animales, leche, cereales, suministro de piensos.
Sector Transporte	Empresas de recogida de envíos y de entrega, empresas de paquetería y de carga fraccionada.

Tabla 1.1 Aplicaciones del VRP a casos reales. Adaptado de The GreenTrip Project

El objetivo del SIL es por lo tanto realizar todas estas actividades de manera coordinada y eficiente, obteniendo un coste total mínimo y garantizando la satisfacción total del cliente mediante el cumplimiento de sus expectativas. Para ello se modeliza el problema a través de su consideración como un VRP, o en ocasiones como un problema de optimización de rutas sobre redes, también conocido como el Problema del Cartero Chino, o *Chinese Postman Problem* (generalmente cuando existen grandes limitaciones en las uniones de los nodos, lo que ocurre con frecuencia en la programación de rutas en ciudades, debido a la existencia de cruces, giros prohibidos, sentidos únicos, etc).

Sin embargo esto no es tan sencillo de realizar, ya que la empresa se encuentra con multitud de problemas a la hora de implantar estos sistemas. Los problemas más típicos de la planificación de las rutas de transporte suelen ser generales para todas las empresas descritas anteriormente. En concreto podemos citar los siguientes:

- Servicios solicitados fuera de hora, lo cual exige una *planificación dinámica* de las rutas.
- Existencia de ventanas de tiempo para cada pedido.
- Diferentes capacidades y PMA (Peso Máximo Autorizado) de los vehículos de la empresa.
- Necesidad de vehículos especiales para determinadas mercancías.
- Restricciones geográficas para determinadas rutas.
- Imposibilidad de utilización de distancias euclídeas (distancia lineal entre dos puntos).
- Restricciones laborales en los horarios de los conductores.
- Velocidades limitadas.
- Incertidumbre en los niveles de tráfico en las rutas.
- Incertidumbre en las posibilidades de carga/descarga en los destinos.

Si a todo esto, añadimos los objetivos de la empresa de perseguir una mejora continua en el servicio, a través de la garantía de cumplimiento de las expectativas de los clientes, la búsqueda de prestigio y reputación en el mercado y la contribución a una utilización eficiente de los recursos y minimización del impacto ambiental, entonces debemos de considerar que alcanzar el óptimo en la planificación es un ideal, aunque sí es posible contribuir a un acercamiento al mismo por medio de una planificación racional y estructurada.

Esta planificación de rutas ha de contener, en concreto, tres actividades complementarias y dependientes entre sí:

- La asignación de cada servicio (sea de recogida o entrega) a un vehículo determinado.
- La secuencia de visitas para todos los vehículos.
- Para cada par consecutivo de visitas, buscar el camino más económico en la red de carreteras.

Para realizar estas actividades es necesario que la solución obtenida respete todas las restricciones impuestas por el planificador o gestor y que la solución sea óptima con respecto a múltiples criterios, como coste total de transporte, satisfacción de clientes, e impacto medioambiental.

En principio este problema de asignación de clientes a los diferentes camiones, está rodeado de la serie de restricciones contempladas anteriormente y en este sentido puede entenderse como un problema de programación lineal entera donde la solución buscada es la referida asignación de clientes a rutas y su secuencia de visitas.

Como cabe pensar, la programación matemática exacta de un modelo de estas características, aunque posible, es ineficiente, dado que el número de soluciones posibles crece de manera exponencial con el número de puntos de servicio. Además el tiempo necesario para obtener las soluciones óptimas puede crecer de forma desproporcionada, tanto más cuantas más restricciones se planteen. Por ello y así se justifica en el capítulo 2, es necesario el desarrollo de modelos heurísticos en base a diferentes niveles de decisión.

Si bien las empresas, en su mayor parte las PYMES, han desarrollado modelos de planificación a medida, para sus respectivas problemáticas, generalmente estos modelos de planificación son de naturaleza intuitiva. Los gestores o planificadores han de manejar tal cantidad de información, que generalmente recurren a la intuición para planificar las rutas, aunque ello no implique siempre un planteamiento de racionalización del proceso y mucho menos de optimización.

En el caso de las empresas de distribución como contrapunto de las empresas puramente de transportes, la situación todavía es más grave. Generalmente estas empresas están habituadas a trabajar con una cartera de clientes más o menos estable, de manera que los puntos de servicio o recogida de mercancías permanecen igualmente estables. Consecuentemente, una de las características del sistema de planificación de rutas, es que éstas son fijas y se sirven con periodicidad diaria. En este sentido, cuando la empresa se plantea servir a nuevos clientes, su decisión se basa más en un criterio de cercanía del cliente potencial a la ruta preestablecida que en un criterio racional y económico. Se trata de una limitación asumida al crecimiento empresarial que podría verse solventada si se tuviese en cuenta una mayor racionalidad en la planificación de rutas. Igualmente ocurre en aquellas empresas de transporte de larga distancia, que ya cuentan con un plan de transportes en el que se contempla las diferentes rutas entre los almacenes centrales de la empresa, así como los horarios de entrada y salida de los vehículos.

En el siguiente ejemplo de la empresa *VIA CONEXIÓN* se muestra al completo su planificación de rutas de transportes entre sus diferentes almacenes para todo el territorio nacional. *VIA CONEXION* realiza en torno al millón y medio de entregas anuales a un abanico de 12.000 poblaciones diferentes. En este sentido cabe señalar que se trata de un proveedor de servicios de transporte a nivel nacional, y que en oferta servicios de entrega urgente. Por ello la importancia de las ventanas de tiempo en el caso de esta empresa es fundamental. Para ello la empresa tiene estructuradas unas líneas de transporte fijo entre

todos y cada uno de sus almacenes, establecida en función de la planificación estratégica de la misma, y de la que deriva la planificación de las rutas de cada uno de los días considerados. Esta empresa, también se caracteriza por aportar la información de todos sus envíos de manera completamente abierta, diferenciándose de la tónica general del sector, donde todas estas actividades permanecen en el más alto secreto empresarial. Por ello lo difícil del acceso a datos reales de empresas para llevar a cabo cualquier tipo de investigación realista sobre la planificación de rutas de transporte. En la figura 1.4 se muestra la información correspondiente a las rutas entre los diferentes almacenes, así como los horarios de las mismas.

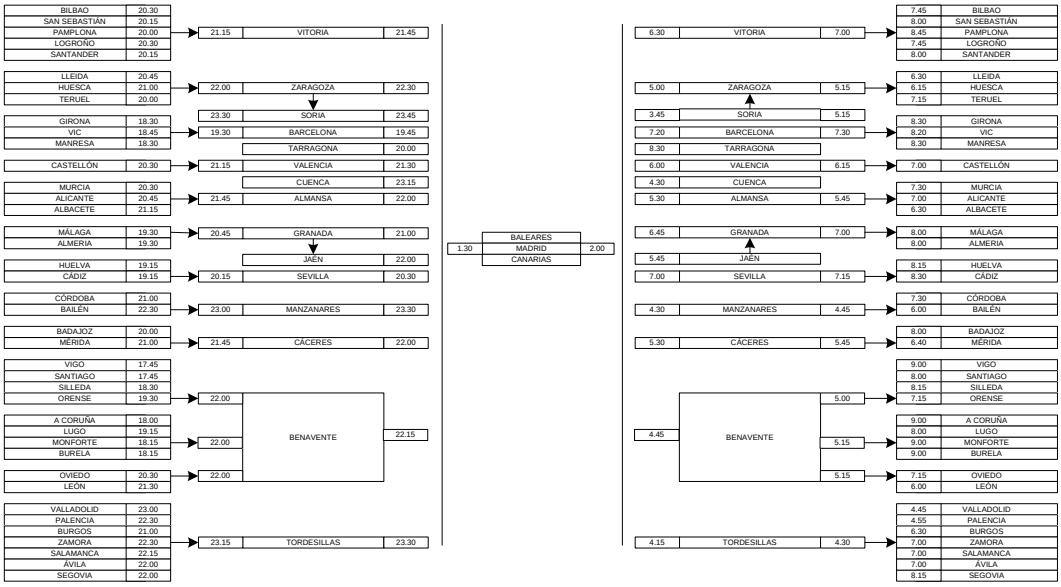


Figura 1.4 Plan de transportes península a larga distancia de la empresa *VIA CONEXIÓN*

En el caso de esta empresa, la integración total y la no dependencia de terceros le permite ofertar a sus clientes elevados índices de satisfacción demostrados a través del cumplimiento de las entregas antes del plazo preestablecido de un 88%, resultando en un nivel más que aceptable en los resultados de esta planificación. Para ello, la planificación de las rutas desde el almacén o plataforma de destino hacia los diferentes clientes receptores de los envíos se realiza por medio de una planificación diaria por parte del delegado de zona, en ocasiones franquiciado, que es quién asume en última instancia el servicio final al cliente.

A continuación se hace especial mención al proceso de planificación de rutas en las PYMES del sector, ya sean de distribución o de transportes, atendiendo tanto al proceso de programación de rutas propiamente dicho, como una consideración sobre los costes en los que la empresa incurre al realizar el servicio de transporte.

1.9 El proceso de planificación

En este apartado se describe con mayor detalle el proceso habitual de gestión de rutas de reparto por parte del empresario o planificador de una PYME. La característica general es que se basa en una gran cantidad de información tácita y un gran conocimiento de la situación, pero no se sigue ningún proceso estructurado o explícito, de manera que muchas veces los resultados se distancian de los óptimos. A pesar de ello en muchos casos la planificación realizada sí obtiene buenos resultados, que generalmente empeoran cuando se incrementa el número de servicios, el número de restricciones, o la mayor complejidad del problema.

Cada día el planificador ha de programar el desarrollo de las rutas. Para ello cuenta con las demandas en firme de diferentes clientes geográficamente dispersos. Estas demandas pueden provenir de diferentes medios, por ejemplo, pedidos recibidos el día anterior, pedidos recibidos durante la noche al almacén, pedidos recibidos por teléfono, por *Internet*, o bien el encargo de reparto de una gran empresa que ha subcontratado el reparto último de las mercancías a la PYME.

El planificador se encuentra con los datos provenientes de esta lista de repartos, generalmente incluyendo información acerca del nombre del cliente, su dirección completa, el número de bultos a repartir o recoger, el volumen de los mismos, su peso y en algunos casos la ventana de tiempo correspondiente.

Además el planificador cuenta con la información de la empresa en cuanto al número de vehículos disponibles, sus capacidades y características, las velocidades medias aproximadas de los mismos, el alcance diario de los vehículos y los costes de cada uno.

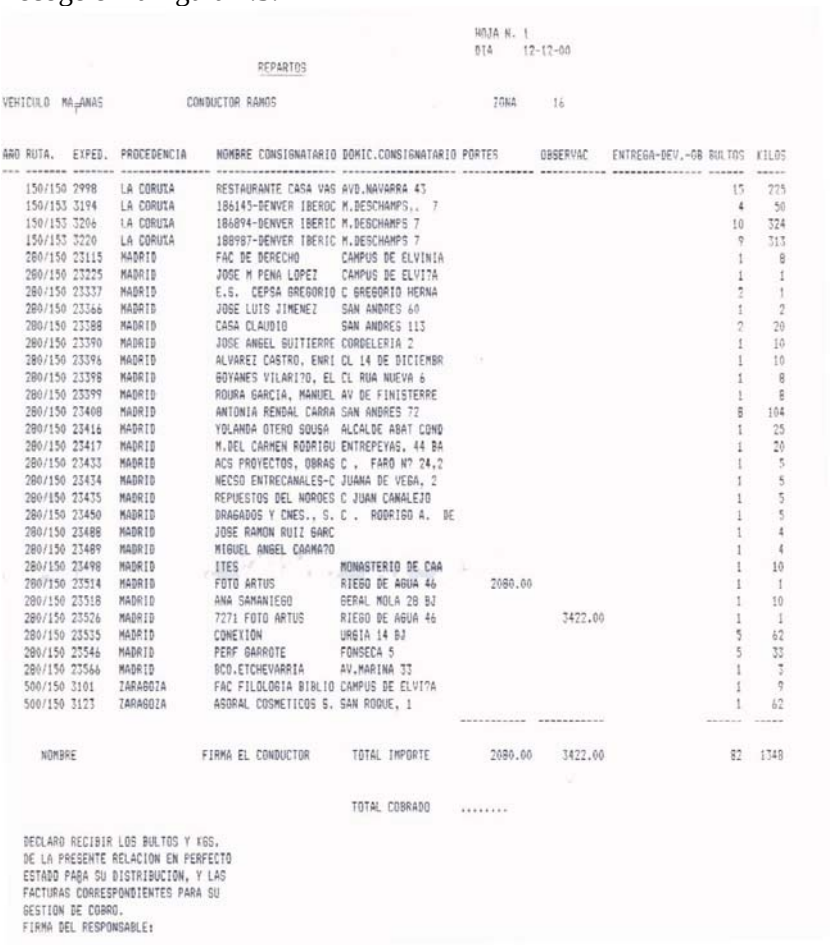
Finalmente el planificador tiene información explícita de las distancias entre diferentes nodos en la red general de carreteras. Esta información suele provenir de las tablas y cuadros de distancias publicadas en guías de carreteras y otras fuentes oficiales. Además el planificador suele contar con su experiencia en cuanto a las distancias y tiempos aproximados de viajes entre las diferentes ciudades o puntos de servicio.

Con todo ello, el trabajo, que ha de ser realizado en el menor tiempo posible, consiste en asignar cada uno de los servicios a los diferentes camiones de manera que sea posible servir todas las órdenes, en el tiempo establecido, e incurriendo en los menores costes totales, garantizando el servicio al cliente.

Normalmente se trata de modelos basados en rutas predeterminadas, ya sean ejes, o zonas preestablecidos. La planificación en base a ejes trata de definir una serie de ciudades preasignadas a la ruta según sea su distancia a una vía o carretera determinada, que configura la columna vertebral de la ruta, mientras que los modelos en base a zonas, asignan los servicios a determinadas zonas para posteriormente asignar un número de vehículos a cada zona. Por ello es habitual ver en los almacenes diferentes espacios indicativos de estas rutas o zonas, donde las mercancías se apilan por separado según la

zona de servicio. Esto indica una primera clasificación de los servicios en el almacén para posteriormente ser asignados a los vehículos correspondientes.

Una vez que el planificador ha realizado la asignación de los servicios a cada ruta, entonces entrega la lista de reparto a cada conductor, que en última instancia son los decisores del orden de reparto seguido entre los clientes. Un ejemplo de estas listas de reparto se recoge en la figura 1.5.



HOJA N. 1
DIA 12-12-00

REPARTOS

VEHICULO MA-ANAS CONDUCTOR RAMOS ZONA 16

AÑO RUTA.	EXPED.	PROCEDENCIA	NOMBRE CONSIGNATARIO	DOMIC.CONSIGNATARIO	PORTES	OBSERVAC	ENTREGA-DEV.-OB	BULTOS	KILOS
150/150	2998	LA CORUÑA	RESTAURANTE CASA VAS	AVD.NAVARRA 43				15	225
150/153	3194	LA CORUÑA	186145-DENVER IBEROC	M.DESCHAMPS,. 7				4	50
150/153	3206	LA CORUÑA	186894-DENVER IBERIC	M.DESCHAMPS 7				10	324
150/153	3220	LA CORUÑA	188987-DENVER IBERIC	M.DESCHAMPS 7				9	313
280/150	23115	MADRID	FAC DE DERECHO	CAMPUS DE ELVIÑIA				1	8
280/150	23225	MADRID	JOSE M PEÑA LOPEZ	CAMPUS DE ELVIÑA				1	1
280/150	23337	MADRID	E.S. CEPSA GREGORIO	C GREGORIO HERNA				2	1
280/150	23366	MADRID	JOSE LUIS JIMENEZ	SAN ANDRES 60				1	2
280/150	23388	MADRID	CASA CLAUDIO	SAN ANDRES 113				2	20
280/150	23390	MADRID	JOSE ANGEL GUITIERRE	CORDELERIA 2				1	10
280/150	23396	MADRID	ALVAREI CASTRO, ENRI	CL 14 DE DICIEMBR				1	10
280/150	23398	MADRID	GOYANES VILARIÑO, EL	CL RUA NUEVA 6				1	8
280/150	23399	MADRID	ROURA GARCIA, MANUEL	AV DE FINISTERRE				1	8
280/150	23408	MADRID	ANTONIA RENDAL CARRA	SAN ANDRES 72				8	104
280/150	23416	MADRID	YOLANDA OTERO SOUSA	ALCALDE ABAT COND				1	25
280/150	23417	MADRID	M.DEL CARMEN RODRIGU	ENTREPEYAS, 44 BA				1	20
280/150	23433	MADRID	ACS PROYECTOS, OBRAS	C . FARO Nº 24,2				1	5
280/150	23434	MADRID	NECSO ENTRECANALES-C	JUANA DE VEGA, 2				1	5
280/150	23435	MADRID	REPUESTOS DEL NOROES	C JUAN CANALEJO				1	5
280/150	23450	MADRID	DRAGADOS Y CNES., S.	C . RODRIGO A. DE				1	5
280/150	23488	MADRID	JOSE RAMON RUIZ GARC					1	4
280/150	23489	MADRID	MIGUEL ANGEL CAAMAÑO					1	4
280/150	23498	MADRID	ITES	MONASTERIO DE CAA				1	10
280/150	23514	MADRID	FOTO ARTUS	RIEGO DE AGUA 46	2080.00			1	1
280/150	23518	MADRID	ANA SAMANIEGO	GERAL MOLA 28 BJ				1	10
280/150	23526	MADRID	7271 FOTO ARTUS	RIEGO DE AGUA 46		3422.00		1	1
280/150	23535	MADRID	CONEXION	URGIA 14 BJ				5	62
280/150	23546	MADRID	PERF GARROTE	FONSECA 5				5	33
280/150	23566	MADRID	BCO.ETCHEVARRIA	AV.MARINA 33				1	7
500/150	3101	ZARAGOZA	FAC FILOLOGIA BIBLIO	CAMPUS DE ELVIÑA				1	9
500/150	3123	ZARAGOZA	ASORAL COSMETICOS S.	SAN ROQUE, 1				1	62
					2080.00	3422.00		82	1748

NOMBRE FIRMA EL CONDUCTOR TOTAL IMPORTE

TOTAL COBRADO .....

DECLARO RECIBIR LOS BULTOS Y KGS.
DE LA PRESENTE RELACION EN PERFECTO
ESTADO PARA SU DISTRIBUCION, Y LAS
FACTURAS CORRESPONDIENTES PARA SU
GESTION DE COBRO.
FIRMA DEL RESPONSABLE:

Figura 1.5 Imagen correspondiente a una hoja de reparto estándar en la que se indican los diferentes destinos, pero no el orden de visita

Después de que comience el reparto por cada camión, no se mantiene ningún contacto con la central salvo posibles contingencias de última hora, por ello no se asumen los nuevos pedidos o servicios que pudiesen surgir en ese día. No se suelen aplicar políticas de planificación dinámica salvo en reducidos casos donde existe un servicio de tipo *urgente*.

Cuando se completan las rutas por cada uno de los vehículos, estos regresan al almacén para repostar y esperar al día siguiente. Cada conductor recoge las notas de entrega y se

las entrega al planificador, para su comprobación. Sin embargo no proporcionan generalmente ninguna información acerca de la calidad o exactitud del servicio realizado, salvo la firma del cliente indicando la recepción.

El sistema seguido por casi la totalidad de las empresas investigadas para realizar este trabajo, es la intuición del planificador, lo cual generalmente proporciona buenos resultados, aunque se ha probado que estos resultados son mejorables a través de la implantación de Sistemas de Información Logística (SIL), generalmente entorno a reducciones del 10% al 20% en los costes totales. Cabe destacar que la planificación intuitiva es tanto más buena cuanto menor es la complejidad del problema. Por ello y ante la creciente evolución del sector, es necesario proporcionar sistemas de ayuda a la toma de decisiones, de forma que estas se adopten en condiciones de mejor información y más racionalidad, persiguiendo alcanzar el óptimo en costes.

Si bien estos SIL no han de servir como un sistema de planificación rígido e incuestionable, sí han de servir como apoyo para la toma de decisiones por parte del planificador, de manera que puede acortar el tiempo de planificación y delegar muchas de las actividades de la planificación al sistema. Además estos SIL han de permitir la flexibilidad necesaria para facilitar cambios por parte del usuario de manera que se puedan realizar análisis de sensibilidad ante los cambios planteados comprobando las variaciones sobre los costes totales.

Por ello el objetivo de los SIL debe consistir en lograr una solución cercana al óptimo donde todas las actividades estén coordinadas. Para ello es necesario partir de una red de carreteras determinada, donde de alguna manera se pueda representar los puntos de servicio, ya sea mediante su localización como ciudades, direcciones o códigos postales. Además es necesario que el SIL sea lo suficientemente rápido como para proporcionar una solución en un breve espacio de tiempo (entre 10 y 15 minutos) y además que permita la modificación y rápida actualización de la solución mediante cualquier cambio realizado por el planificador.

En la medida de lo posible se ha de vincular este sistema con las restantes aplicaciones informáticas de la empresa mediante el intercambio de la información necesaria, ya sea con las actividades previas, como por ejemplo listados de servicios para un día en concreto, o pedidos de clientes a servir, etc. y también para las actividades posteriores, como por ejemplo los sistemas de seguimiento de mercancías, o los sistemas de imputación de costes.

En los dos apartados siguientes se hace especial consideración a los costes que han de contemplarse a la hora de establecer la planificación de rutas. Como se comentaba al principio, el objetivo último de la planificación es incurrir en los menores costes posibles garantizando el servicio al cliente y su total satisfacción.

1.10 Costes fijos en la distribución

La empresa de distribución o transportes cuenta con múltiples medios tanto materiales como humanos destinados a la planificación y ejecución de las rutas, que permanecen invariables ante los cambios en el número de pedidos repartidos en cada día. Generalmente estos costes fijos hacen referencia a los costes estructurales de la empresa y que se corresponden con las inversiones realizadas en infraestructuras para soportar todo el sistema. A continuación se mencionan los costes más comunes en el sector de distribución.

En primer lugar, la empresa ha de contar necesariamente con unas instalaciones donde se albergan las mercancías repartidas. La inversión en el inmovilizado da lugar a las correspondientes amortizaciones, que generalmente siguen un criterio de tipo lineal o temporal, por lo que generan un coste fijo anual que ha de ser soportado haya actividad o no.

En el almacén generalmente se poseen medios e instalaciones para la distribución y clasificación de los envíos, ya sea manual o bien automatizada, así como sistemas de control e identificación, generalmente por códigos de barras o radiofrecuencia. Además el almacén ha de estar dotado de las oportunas bahías de carga para facilitar el acceso a los camiones, e incluso las instalaciones de carga y descarga oportunas para facilitar todo el proceso.

En cuanto a los recursos de personal, es necesario tanto el personal administrativo encargado de supervisar y controlar todo el sistema y que se hallan bajo la dirección del planificador, como del personal de almacén encargado de ejecutar las acciones previstas por el planificador, consistentes en la carga y descarga de las mercancías en los vehículos y su distribución interna en la planta.

Finalmente la empresa cuenta con la flota de camiones o vehículos de reparto, así como con los conductores de los mismos. Habitualmente estos vehículos componen una flota heterogénea que permite la utilización de uno u otro vehículo según las necesidades de cada momento. Los costes de personal fijo en la empresa, así como los costes de amortización del vehículo suelen ser considerados también como costes fijos anuales.

Además la empresa puede contar con talleres propios y garajes donde se realizan las tareas más habituales de mantenimiento y pequeñas reparaciones en los vehículos, a manos de algún mecánico contratado directamente por la empresa. En ocasiones, algunas de estas empresas tienen incluso el suministro de combustible propio, así como túneles de lavado automático.

1.11 Costes variables en la distribución

En este apartado se hace referencia a los costes variables de la empresa, que generalmente derivan de los costes por cada kilómetro recorrido, y que se corresponden con el segundo componente del coste total por kilómetro.

Cabe mencionar que cada vez más tiende a incrementarse la proporción de costes variables sobre costes fijos en el caso de las empresas de transporte. De esta manera, y motivado por las razones expuestas al principio de este capítulo, cada vez tiene una mayor importancia los costes variables por cada kilómetro recorrido, que no los costes fijos de las empresas de transporte. Esta circunstancia no es propia únicamente de este sector, ya que en la mayor parte de las actividades empresariales existe una tendencia a convertir costes fijos en variables a través de múltiples herramientas de gestión. Entre ellas, por ejemplo la subcontratación de servicios de transporte a terceros operadores o autónomos, o los sistemas de franquicias. En este sentido, la necesidad de reducir al mínimo los costes por kilometraje todavía apuntala más la necesidad de buscar sistemas eficientes y eficaces de planificación de rutas de transporte que minimicen la totalidad de los costes manteniendo unos mínimos niveles de calidad en el servicio.

En la siguiente tabla se recogen de manera más detallada los costes variables imputados a cada Kilómetro recorrido.

COSTE DEL CONDUCTOR
Coste de sueldos y salarios. Se debe tener en cuenta el número máximo de horas de conducción permitidas legalmente al día. **Costes de manutención**: Incluyen comidas y estancias, en caso de desplazamiento al extranjero se debe tener en cuenta las dietas correspondientes.
COSTES DIRECTOS DEL VEHÍCULO
Consumo de gasoil. Directamente proporcionales a la distancia y al tiempo. **Consumo de aceite**. Directamente proporcionales a las distancias recorridas. **Consumo de neumáticos**. Directamente proporcionales a las distancias recorridas y al tipo de carreteras utilizadas. **Pastillas de frenos**. Directamente proporcionales a las distancias recorridas y al tipo de carreteras utilizadas. **Otros consumos:** Mantenimiento: aceites utilizados (Valvoline, líquido de frenos, de servo dirección etc.), elementos que sufren desgaste con el uso; embragues, bombillas, correas, etc. Se deberían analizar e incluir en mantenimiento preventivo y aplicar una tasa por Km. dado que están relacionados con las distancias recorridas.

Tabla 1.2 Desglose de componentes de costes por Km recorrido

Igualmente cabe mencionar la necesidad de determinadas empresas de acudir a la subcontratación de determinados servicios. Esto es habitual en los períodos en los que la empresa se encuentra con puntas de demanda, generalmente de difícil previsión, o bien si

estas están previstas, puede ser habitual subcontratarlas para contar con una mayor flexibilidad. La subcontratación de estos servicios ha de ser un último recurso para la empresa, aunque cada vez más se percibe como un factor adicional de la tan necesitada flexibilidad en las empresas.

También es habitual recurrir a la subcontratación para el servicio de lo que podríamos denominar "*outliers*", o servicios aislados. Existen en el sector determinados profesionales que se dedican a servir única y exclusivamente estas zonas lejanas, poco habituales, de manera que acuden a las empresas de las zonas más concentradas para recoger en las diferentes empresas todos los envíos destinados a las zonas más lejanas y así rentabilizar el viaje. Este hecho hace referencia a lo comentado en el primer epígrafe de este capítulo donde se contempla la especialización de las PYMES en zonas poco rentables para la gran empresa, como una de las posibles formas de supervivencia ante la creciente internacionalización y concentración del sector.

Tanto la primera como la segunda situación, implican para la empresa la existencia de costes variables y que generalmente se relacionan con la distancia de servicio de cada uno de los pedidos o entregas. En este sentido también se pueden obtener ventajas debido a la racionalización del proceso de externalización, dado que en muchas situaciones puede ser interesante reconsiderar las opciones de subcontratación de determinados servicios.

En el siguiente apartado se desarrollan las soluciones existentes para la planificación de rutas de transporte, centrando la atención en dos campos, por un lado las aplicaciones desarrolladas para la resolución del TSP y en segundo lugar aquellas planteadas para la resolución del VRP.

2 Evolución de las soluciones para la planificación de rutas de transporte

Tal y como se comentaba en el apartado anterior, cuando el planificador se encuentra ante un problema de programación de rutas o de gestión de una flota de vehículos, generalmente apuesta por su intuición. Sin embargo en los últimos años se han desarrollado soluciones informáticas que permite la realización del proceso de planificación en condiciones de mayor información y mayor racionalidad. En este apartado se discute y se contrastan las soluciones desarrolladas por los diferentes autores o empresas de software.

Los usuarios generalmente se encuentran con una doble vertiente a la hora de decidir qué tipo de solución es más conveniente. Por un lado tienen la opción de utilizar una aplicación informática comercializada de las existentes en el mercado y por otro lado pueden desarrollar su propia solución para su caso concreto, lo cual generalmente provoca que este tipo de soluciones solamente sean válidas en esas circunstancias para las que han sido concebidas, resultando imposible su aplicación en otros casos, además de suponer un coste mucho más elevado que los paquetes informáticos comercializados. En este apartado se comentan en mayor profundidad las soluciones comercializadas.

Dentro de las soluciones existentes y comercializadas nos encontramos con dos tipos de aplicaciones. Aquellas destinadas a la investigación operativa para la resolución de problemas en el campo teórico y que permiten alcanzar elevadas tasas de optimización y aquellas que van enfocadas a las empresas para la resolución de problemas reales.

Dentro de las herramientas más teóricas, se incluyen las derivadas de los algoritmos necesarios para resolver los problemas TSP y que generalmente consisten en versiones muy sencillas, generalmente de demostración de algún algoritmo concreto.

1.12 Herramientas para la resolución del TSP

Las aplicaciones de TSPs son de carácter muy variado. Existen aplicaciones de simple demostración de los algoritmos propuestos por cada uno de los autores, capaces de generar ejemplos de TSPs y de resolver dichos casos con unas rutinas predeterminadas y por otro lado existen aquellas aplicaciones enfocadas a la resolución de cualquier tipo de TSP, generalmente aplicadas a campos como la fabricación de tarjetas de circuitos integrados donde el número de nodos es muy elevado.

A continuación se describen algunas de estas herramientas desarrolladas por diferentes autores para la resolución de TSPs. La característica general es que todos ellos trabajan con distancias euclídeas calculadas a partir de las coordenadas de los nodos.

TSP SOLVER[3] desarrollado por Myagkih, Savelev y Kureichik. Se trata de un algoritmo basado en el planteamiento de los algoritmos genéticos más sencillos. Consigue buenos resultados pero no puede trabajar con más de 100 nodos.

DAKIN PATHFINDER[4] desarrollado por Dakin. Se trata de un algoritmo basado en dos fases. Primero opera con algoritmos rápidos de construcción de rutas y posteriormente realiza una postoptimización. Es un algoritmo más rápido que el anterior, pero que también trabaja con problemas de pocos nodos, generalmente de menos de 100 nodos.

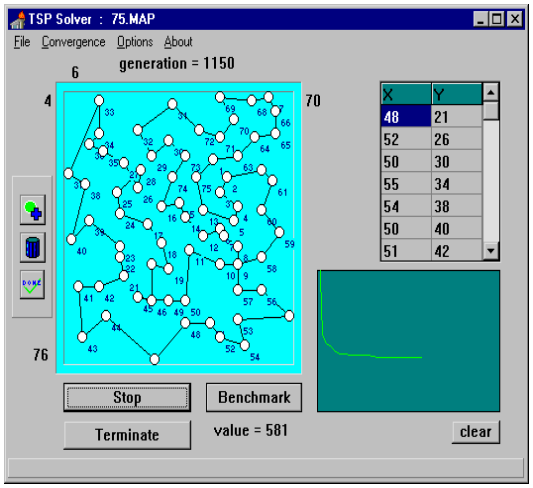


Figura 1.6 TSP SOLVER: En este ejemplo se intenta resolver el problema denominado Eilon75.

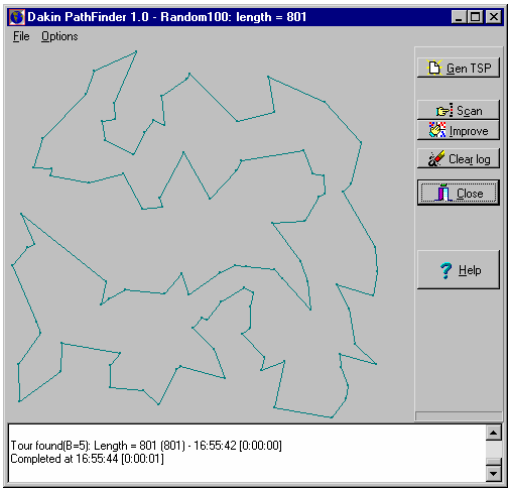


Figura 1.7 DAKIN PATHFINDER

En este grupo de planteamientos destaca el código escrito por Ch'vatal, et al.[5] denominado CONCORDE, que es capaz de resolver de forma óptima problemas de hasta 13.000 nodos y de manera aproximada puede llegar a proporcionar soluciones cercanas al 2% sobre el óptimo para problemas de hasta 10 millones de nodos. Este software permite establecer cuál es el recorrido mínimo entre un grupo de puntos o nodos que han de ser visitados, sin ningún tipo de restricción adicional. Para ello tiene en cuenta multitud de algoritmos matemáticos y rutinas para obtener dicha solución óptima.

Se trata en concreto de una librería de algoritmos sencillos, incluyendo todas las técnicas de optimización actuales, desde los algoritmos heurísticos de primera generación para la construcción de rutas, hasta las últimas herramientas de programación matemática. Son rutinas que han de ser ejecutadas bajo un software de optimización, como por ejemplo el CPLEX. Los códigos están disponibles en línea a través de *Internet* para su utilización en investigación operativa. Para aplicaciones empresariales, es necesario abonar los derechos de utilización.

[3] Véase http://web.cps.msu.edu/
[4] Véase http://www.pcug.org.au/~dakin/tsp.htm
[5] Véase http://www.math.princeton.edu/tsp/concorde.html

A continuación se muestra uno de los últimos problemas resueltos hasta el óptimo por estos autores. En este ejemplo se muestra la mejor ruta para la unión de las 13.509 ciudades más importantes de Estados Unidos, conseguida a través de la implementación del software CONCORDE.

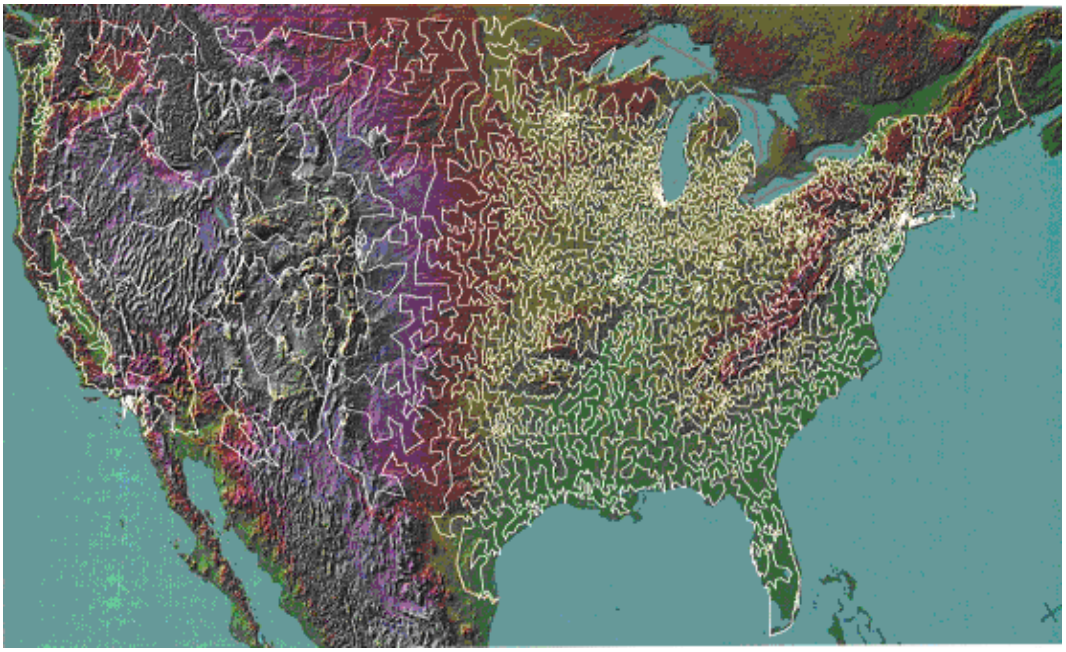


Figura 1.8 Solución al problema usa13509 del TSPLIB obtenida por Ch'vatal et al. utilizando el algoritmo CONCORDE.

1.13 Herramientas para la resolución de VRPs

Dentro de las aplicaciones del VRP, también existe la dualidad planteada anteriormente. Por un lado existen aquellos desarrollos informáticos enfocados a la resolución de los casos más teóricos planteados por la literatura y que generalmente sirven de demostración de los algoritmos propuestos por los autores y por otro lado, existen aquellos programas informáticos dirigidos a la solución de problemas reales de VRP, generalmente válidos para el caso específico del VRPTW.

Mayoritariamente los programas de aplicación práctica son planteamientos *ad hoc* que han de venir solicitados por las empresas a empresas especializadas de desarrollo de software, dado que exigen la digitalización de la zona donde opera la empresa. Es por ello que, a pesar de las grandísimas ventajas que reporta, el número de experiencias en este campo son de ámbito mucho más reducido. Algunas de estas experiencias se describen en la revisión literaria recogida en el capítulo 2.

Son notorias las investigaciones realizadas por el instituto *Transport en Logistiek Institute* [6]en su revista acerca de las soluciones sobre tecnologías de la información para la

[6] Véase http://www.vtenl.nl/

logística. En 1991 las conclusiones obtenidas indicaban que las soluciones informáticas existentes en aquella época no eran tan eficientes como los humanos para calcular las rutas. Sin embargo a finales de 1993, en su segunda investigación sobre el caso, mostraban como el software disponible había evolucionado hasta el punto de poder resolver problemas de VRP, con unos resultados cercanos al óptimo, pero para problemas muy sencillos, generalmente derivados de los casos propuestos por la literatura. El estudio realizado establecía que las soluciones de entonces todavía no estaban preparadas para trabajar con el elevado número de restricciones y variedad de problemas que pueden darse en la realidad.

A finales de 1999 concluía el desarrollo de The GreenTrip Project con unos resultados prometedores en cuanto a la resolución de los problemas tipo de la literatura, aunque como se menciona en el párrafo anterior, basado en distancias euclídeas. Como resultados prácticos de este proyecto se han desarrollado dos aplicaciones informáticas diferentes, comentadas en los siguientes apartados.

Las evoluciones tanto en velocidades de computación como en las aplicaciones desarrolladas en múltiples campos de la informática, han permitido el desarrollo de potentes herramientas de planificación de rutas de transporte. En este sentido la integración de diferentes tecnologías, como son los sistemas de información geográfica, los sistemas de decisión empresarial, los algoritmos de optimización, y la posibilidad de visualizar y realizar seguimientos de las flotas de vehículos, permiten hoy día el que la planificación empresarial sea una cuestión donde ya no es necesaria la experiencia o el instinto del planificador. Hoy día es posible mantener un contacto constante con todos los vehículos de una flota de transportes, y no solamente eso, sino que gracias a los sistemas de seguimiento (*Qualcomm, HighwayMaster, AMSC*) es posible saber las duraciones de las paradas de los conductores, dónde paran, los resultados de velocidades medias, consumos, las posiciones de los vehículos en cada momento, la localización de los puntos de servicio, o de recogida, la planificación dinámica, y en resumen, la automatización de todas las tareas requeridas para la planificación de rutas de transporte.

Igualmente se han desarrollado los sistemas de navegación (*Alpine, BMW, Oldsmobile y Toyota*) que proporcionan al conductor un inmejorable copiloto que anuncia con segundos de antelación las diferentes direcciones o instrucciones que este ha de seguir para llegar al destino seleccionado. Permiten el establecimiento de rutas entre dos o más puntos a través de la selección del camino más corto, o más rápido, visualizando al mismo tiempo estas rutas en planos o mapas digitales de la zona en cuestión. Estos sistemas son los combinados entre los sistemas de seguimiento fundamentados en la tecnología GPS, y los sistemas de información geográfica GIS.

Actualmente se presenta la oportunidad de integrar todos estos sistemas en productos de software que sean fácilmente adaptables a las necesidades de cada empresa. Si bien todavía hoy no existe ningún producto que contemple todos los campos, sí existen productos enfocados a cada una de esas necesidades en el mundo de la planificación de rutas de transportes.

La mayor parte de los productos existentes en el mercado consisten en *DSS*, o *Decision Support Systems*, para el apoyo de las decisiones que toma el planificador, y en algunos casos para el control de estas decisiones. Generalmente estos productos contemplan un soporte de información geográfica a través de las tecnologías de *MapInfos, ArcInfos*, y *GDT*, así como una interfaz gráfica desde donde el usuario introduce todos los datos relativos a los envíos o entregas, a partir de la cuál el algoritmo matemático de planificación devuelve los resultados a modo de rutas de vehículos en las que se ha programado la secuencia de servicios a realizar por cada vehículo, respetando en todo momento las restricciones impuestas por el planificador.

Hasta el año 2000, la línea de investigación operativa enfocada a la resolución del problema de planificación de rutas de transporte, había permanecido dentro de las facultades y escuelas de ingeniería. Sin embargo en los últimos años, se ha producido un enorme cambio que afecta a todas las áreas dedicadas al problema del VRP. En este sentido las empresas *ESRI* e *ILOG* han abierto sus librerías de códigos a la comunidad científica y al mundo empresarial para el desarrollo de aplicaciones combinadas a través del pago de algún tipo de royalty o canon.

ILOG Dispatcher es una librería de instrucciones programadas en C++ para trabajar sobre el *ILOG Solver*, que consiste en una plataforma de programación con restricciones muy potente, y que actualmente se está implantando en combinación con procedimientos de búsqueda local, en algunos grupos de investigación operativa dedicada al problema del VRPTW, tal y como se comenta más adelante en el capítulo 2. De Backer es el máximo defensor de la aplicación de este tipo de técnicas para la resolución del VRPTW, y al mismo tiempo es el Senior Project Manager del *Dispatcher*.

En el caso de *ESRI*, esta empresa proporciona su producto *NetEngine*, consistente en una librería de algoritmos de optimización de redes, que puede ser integrado a través de la programación en C, generando las correspondientes *APIs (Application Program Interfaces),* o bien a través de *Visual Basic*, regenerando los procedimientos contenidos en la misma. Igualmente *ESRI* suministra todo el software GIS para integrar los sistemas, y crear productos de alto valor añadido destinados a la planificación de rutas de transporte.

En los últimos años, las aplicaciones originalmente creadas para entornos de oficina y destinadas al seguimiento de flotas de vehículos, se han volcado en la integración de estos sistemas en *Internet*, de manera que los usuarios pueden obtener información del seguimiento de sus vehículos a través de *Internet*, desde cualquier lugar del mundo, y no solamente desde la oficina.
El mundo de las tecnologías inalámbricas está de auge en el sector del transporte. El abaratamiento de estas tecnologías está empujando a muchas empresas a implantar este tipo de sistemas en sus actividades de planificación. Algunos ejemplos de este tipo de software son *Mobile Link Freight* y *Mobile Link Tracker* de *Descartes*, a través de los que es posible transferir todo tipo de información entre el vehículo y la oficina de manera

que es posible reprogramar las rutas existentes en el caso de que ocurriera cualquier contingencia en el desarrollo de las mismas.

Igualmente se pone de manifiesto la necesidad de reducir los tiempos de computación y los recursos de información necesarios para las empresas de transporte. En este sentido la empresa *GeoCom USA* está desarrollando aplicaciones de cliente-servidor, de manera que cada vehículo se conecta sistemáticamente al servidor de la empresa para no almacenar tanta información y poder procesar los datos con mayor rapidez.

Para comprender el abanico de las diferentes soluciones informáticas dedicadas a la planificación de rutas cabe destacar los trabajos realizados por Hall (2002) quien periódicamente publica los resultados sobre las encuestas destinadas a los proveedores de estas soluciones informáticas para la planificación de rutas de transporte[7]. En este sentido, uno de sus últimos trabajos es la encuesta del 2002, publicada en Febrero, dentro de la publicación *OR/MS Today* de *INFORMS*. De estas encuestas periódicas se desprenden además de los datos de las últimas plataformas de software desarrolladas para la resolución de los procesos de planificación de rutas de transporte, una serie de cuestiones de vital importancia en el mundo de la logística, y que se pueden revisar en uno de sus últimos trabajos Hall (1999). A continuacións se comentan algunos de estos resultados.

La mayor parte de los algoritmos utilizados establecen la asignación de los nodos a los diferentes almacenes, y para cada uno de los almacenes se desarrollan las correspondientes rutas de transporte realizadas por sendos vehículos. Esta asignación se realiza de forma paralela, y no secuencial, de manera que los resultados se mejoran. La mayor parte también expresan que las distancias tenidas en cuenta se fundamentan en las distancias reales a nivel de callejero, de forma que son necesarios realizar exhaustivos cálculos para computar todas las distancias necesarias en una determinada planificación.

Igualmente contemplan las dos posibilidades de planificación estática, antes de que partan los vehículos, y la planificación dinámica, cuando los vehículos ya están cubriendo las rutas.

Una ventaja indudable de este tipo de software es la posibilidad de modificar las rutas a través de los procedimientos de *arrastre* sobre la interfaz, de manera que el usuario puede aplicar cambios directamente sobre la misma, de manera muy simple. La mayor parte de los productos informáticos contemplan esta posibilidad, ya que fundamentan sus interfaces gráficas en mapas digitales elaborados por otras compañías, generalmente de *Navigation Technologies*, así como los mapas proporcionados por fuentes públicas como es el caso de los mapas *Tiger* del Gobierno de los EE.UU. En este sentido existe una gran dependencia de los desarrolladores de software sobre los diseñadores de mapas digitales, aunque en el caso de algunos mercados, como los Estados Unidos, existen multitud de proveedores de estos mapas digitales a precios muy económicos, no siendo así el caso de

[7] Ver anexo I

otras regiones, como Europa, y en particular, las zonas más periféricas, como es el caso de Galicia.

Otro dato a destacar es que la mayor parte de los proveedores de software se habían especializado en un segmento determinado, aunque también existían herramientas no diferenciadas para cualquier sector. En este sentido, las empresas *Trapeze, RouteLogic* (*Compass*) y *VersaTrans* se centran en la industria de tránsitos; *UPS Logistics* se especializa en la industria de distribución de bebidas y comidas y la empresa *CAPS Logistics* concentra su actividad en la distribución de bienes manufacturados. Por otro lado, empresas como *MicroAnalytics (TruckSTOPS)* y *ESRI* (*ArcLogistics Route*) desarrollan productos estándar destinados a un mercado más amplio y a precios más reducidos.

A continuación se detallan algunas de las aplicaciones informáticas[8] contempladas en la encuesta de Hall (2002) para la resolución de problemas logísticos de tipo VRP. En concreto se mencionan los casos del *SINTEF Spider*, *ILOG Dispatcher* (ambas derivadas de los resultados de The GreenTrip Project), *OPTRACK 4* de la empresa *OPTRAK*, el *ArcLogistics Route* de *ESRI*, el *SUPER IFMAP* de *IBM*, y por último los desarrollos de *DPS*, actualmente integrada en el grupo *PriceWaterhouse Coopers*.

1.13.1 SINTEF Spider

Spider es la aplicación para la planificación óptima de redes de transporte desarrollada por el *SINTEF* como resultado del proyecto de *The GreenTrip* financiado por la UE. Permite su adaptación a cualquier situación y contempla las necesidades más demandadas por las empresas, entre ellas la planificación dinámica de rutas.

El *Spider* contiene tres módulos diferentes.

- **Administrador del Plan**: Este módulo contiene las estructuras de datos y funciones necesarias para administrar los planes y los datos de entrada.
- **Motor de optimización**: Es el componente inicial para elaborar planes iniciales y optimizarlos a través de un proceso interactivo.
- **Topología**: el módulo de topología calcula la información relativa al tiempo y distancia para utilizarla en el módulo de optimización. La información se puede basar en tablas, cálculos realizados con coordenadas (Distancias Euclídeas) o bien en cálculos exactos basados en información de mapas de carreteras digitales.

En 1999 se lanza al mercado la primera versión que trabaja sobre plataformas *Windows*. El lenguaje de programación empleado es C++ y se ha utilizado *Visual Basic* para la elaboración de las interfaces.

A continuación se desarrolla con más detalles las especificaciones de este programa:
RUTAS. Los datos iniciales para la elaboración de las rutas son los siguientes:

[8] Para mayor información sobre estas soluciones informáticas cabe destacar los trabajos de Toth y Vigo (2002).

- **Vehículos**: Están definidos por su *capacidad*, que puede venir expresada en base a cualquier número de dimensiones. Por ejemplo, peso, volumen, número de pallets, etc. de manera que la capacidad de cada vehículo no puede ser violada en ningún momento. Estas dimensiones y la restricción vienen determinada por el usuario.
- El **conductor**.
- La **ruta**: representa un cierto intervalo de tiempo en el que un vehículo está disponible para realizar el transporte. La ruta ha de ser asignada a un vehículo y a un conductor. Además existe una localidad de partida, desde la que comienza la ruta en un *momento inicial de partida* y también existe una *localidad de llegada*, con un *tiempo límite de llegada*.

PEDIDOS

- Un **pedido** representa la demanda de transporte de determinados productos. Cada pedido se define por una tarea a realizar (de recogida o entrega) y por una localidad en la que se realiza el servicio. Además el pedido tiene un tamaño determinado, expresado en las mismas medidas que la dimensión de los vehículos.

PLANES

- Una vez que se han definido los pedidos servidos en una ruta, es el momento de elaborar el plan. Este plan consiste en la secuencia de tareas realizadas en cada ruta, determinando qué mercancías y qué ruta se asigna a cada vehículo. Si se violan las capacidades de la empresa, entonces habrá pedidos que no se puedan servir.
- Se incluye un módulo de realización de informes para el planificador, a través del cuál se pueden realizar diferentes consultas.

RESTRICCIONES

SPIDER permite la inclusión de restricciones en el planteamiento básico. Las posibilidades contempladas son:

- Restricciones de ventanas de tiempo para cada una de las tareas realizadas para cada pedido.
- Restricciones de capacidad para cada vehículo.
- Restricciones de compatibilidad: es posible asignar de manera determinista ciertos vehículos a determinadas zonas, o ciertos pedidos a determinados conductores.

OBJETIVOS

A través de la definición de los objetivos, se introduce información acerca de cómo se calcula el coste de un determinado plan. Para ello se tiene en cuenta el coste por distancia, por tiempo, o ambos. Contiene además un componente que permite realizar un análisis de sensibilidad atendiendo a las variaciones que se producen entre cualquier modificación realizada sobre las rutas. También se puede establecer una cierta jerarquía en los pedidos que se han de servir para diferenciar cuáles son más o menos importantes.

PLANIFICACIÓN DINÁMICA

Permite una planificación dinámica de las rutas, a través de la actualización en tiempo real de la evolución de las rutas de manera que es posible modificar los planes sobre la marcha, retirar servicios o incluso añadir otros nuevos. Los planes se pueden guardar, editar y modificar a gusto del usuario. Es posible incluso bloquear parte del plan, por ejemplo una ruta que se ha predeterminado.

TOPOLOGÍA

Hay dos posibilidades, la topología euclídea y la topología de carreteras.

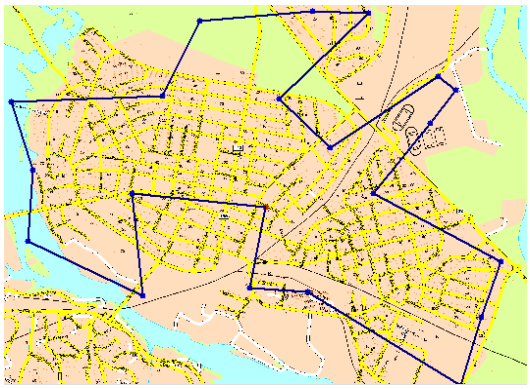


Figura 1.9 Representación de distancias euclídeas

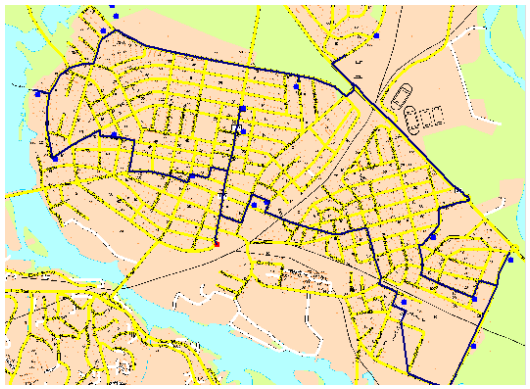


Figura 1.10 Representación de distancias reales

La primera calcula las distancias, tiempos y costes partiendo de las coordenadas de las ciudades, mientras que la segunda tiene en cuenta el camino más corto en una red de carreteras y cruces. Para ello se tienen en cuenta las siguientes características.

- Las carreteras tienen una longitud determinada y un límite de velocidad para calcular la distancia y el tiempo de viaje.
- Las carreteras pueden ser de un solo sentido
- Las carreteras pueden tener barreras físicas para algún vehículo.
- Los cruces pueden tener restricciones de giro, impidiendo determinados movimientos.
- Puede haber restricciones de altura, longitud o peso para determinadas carreteras.

Spider puede leer los mapas de carreteras a través del formato SOSI. (Las aplicaciones desarrolladas hasta el momento se basan en la red de carreteras de Noruega, proporcionada por *Transport Telematik, A.S.*)

El coste de la planificación de una ruta se puede medir utilizando una métrica determinada, por ejemplo como combinación de tiempo y distancia (en los casos descritos se habla de 83 euros por hora). Por ello se puede solicitar la optimización mediante los caminos más cortos o bien los más rápidos. El cálculo del camino se realiza a través de un algoritmo muy eficiente consistente en una variación del conocido algoritmo de Dijkstra.

Los nodos de servicio se pueden establecer bien con un par de coordenadas, o bien con una dirección determinada. Estos últimos han de ser transformados en coordenadas a través de una guía de direcciones. El cruce más cercano a las coordenadas es utilizado como el punto de conexión del nodo en la red de carreteras. Opcionalmente se puede definir un código de carretera (correspondiente al nombre de la calle), el cuál debe de corresponderse con una de las calles en el cruce o intersección.

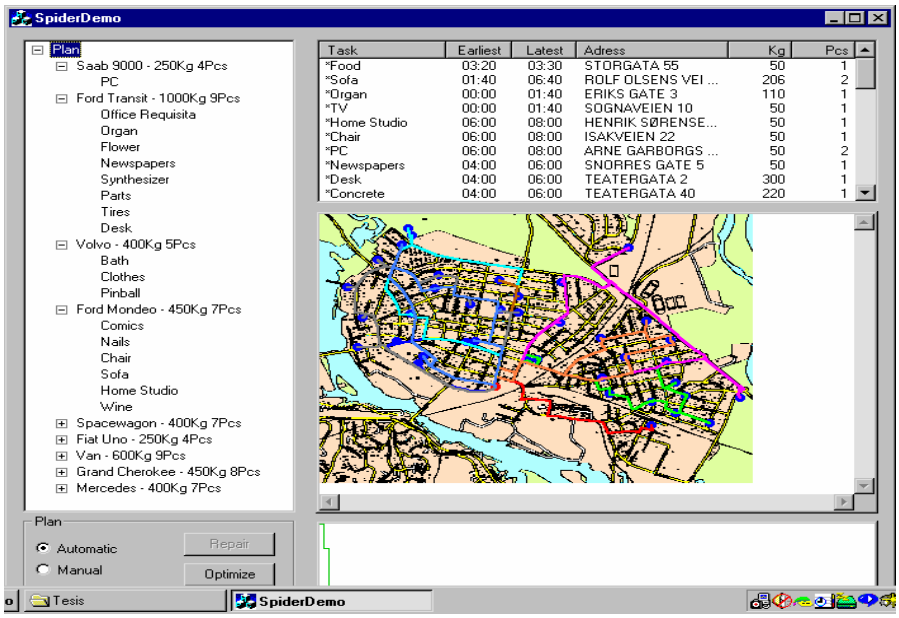


Figura 1.11 Interfaz del programa SPIDER del SINTEF (www.oslo.sintef.no/spider/)
Permite optimizar rutas de transporte de una manera flexible teniendo en cuenta múltiples restricciones (red de carreteras, vehículos diferentes, ventanas de tiempo, restricciones de capacidad, pesos, etc.)

1.13.2 ILOG Dispatcher

Se trata de una aplicación para la gestión de rutas de transporte basada en un motor de optimización denominado *ILOG Optimization Suite*. . Esta aplicación también surge del proyecto The GreenTrip Project y es desarrollado por uno de sus socios colaboradores, la compañía *ILOG*. Su primera aplicación es en la empresa *TollPost Globe* de Noruega,

dedicada al sector de transportes. En la revisión bibliográfica se menciona este caso con mayor detalle.

El programa trabaja con una serie de objetos predefinidos que el usuario selecciona para planificar cada caso concreto. De esta manera el usuario puede adaptar la aplicación para resolver sus propios problemas de rutas de una manera sencilla en términos de visitas y vehículos disponibles.

Para obtener una mayor adaptabilidad, la aplicación tiene la posibilidad de desarrollar una serie de restricciones, tales como una capacidad restringida de la flota, ventanas de tiempo de las entregas o recogidas, regulación de horarios de trabajadores. Además:

- Permite la resolución de VRPs con múltiples depósitos, de manera que la aplicación selecciona el mejor depósito para cada servicio realizado.
- Incluye la posibilidad de actualización en tiempo real, de manera que se pueden añadir visitas cuando las rutas ya se están ejecutando, teniendo en cuenta unas velocidades medias.
- Incluye la posibilidad de contemplar tanto recogidas como entregas.

De esta manera el programa proporciona la solución de mínimo coste teniendo en cuenta distancias, tiempos o ambos.

Figura 1.12 Interfaz del ILOG Dispatcher.

Los algoritmos utilizados para obtener esta optimización constan de dos fases o etapas. En primer lugar, el algoritmo genera una solución posible por medio de métodos predeterminados y, a continuación, aplica técnicas de búsqueda local para obtener mejoras a través de la búsqueda tabú.
Es necesario establecer una elección entre mantener una generalidad del programa basándose en distancias euclídeas entre poblaciones –de manera que basta con dibujarlas sobre un mapa– o bien utilizar bases cartográficas del territorio.

La primera elección implica que la solución proporcionada por el algoritmo programado se consigue de manera muy rápida y sencilla, pero muchos de los arcos entre ciudades

planteados serán imposibles, bien porque existen barreras naturales entre las ciudades, como montañas, ríos o lagos, bien porque las carreteras nunca son rectilíneas, o bien porque simplemente no existe tal unión en la red de carreteras.

La segunda opción, a pesar de ser mucho más laboriosa, sí permite la consideración de las distancias reales entre las ciudades, así como la posibilidad o imposibilidad de unión entre cualquier par de las mismas. Además permite una representación veraz y fiable, tal y como se representa en la figura 1.12.

1.13.3 OPTRAK 4

La empresa *Optrak* también ofrece sus propias soluciones para la planificación de rutas de transporte, de manera similar a las anteriores empresas. En concreto ofrece su programa de planificación de rutas denominado *Optrak 4*. Este software se caracteriza, al igual que los anteriores, por su enorme flexibilidad e interactividad entre el usuario y el problema a resolver.

El software permite a través de la introducción de datos de la empresa, realizar la planificación de las rutas de transporte para cada uno de los depósitos de la empresa. Esta aplicación permite la utilización conjunta de los algoritmos de optimización para el caso de múltiples depósitos. La empresa no suministra información acerca de los algoritmos empleados para realizar los cálculos.

La información se centra en el uso de bases de datos con la información relevante de cada ruta de transporte, incluyendo datos sobre ventanas de tiempo y los recorridos de cada vehículo.

Trip T0001

Trip: T0001		Driver: Bir-artic-0		Feas: OK			
Customer	**Order**	**Result**	**Type**	**Weight**	**Volume**	**Arrive**	**Load.**
Birmingham				5451	272	06:00	0:22
29B454616	63935		del	664	30	08:35	0:18
58B323533	05110		del	540	52	08:53	0:17
32A148850	72982		del	1065	60	10:04	0:23
76F335994	43299		del	534	21	10:34	0:17
94B760822	85331		del	330	21	12:23	0:14
61B673080	33512		del	2318	88	13:41	0:39
Birmingham				0	0	15:43	0:00
Total	**6**	**Delivered**		**5451**	**272**		**2:30**
		Collected		**0**	**0**		
		Peak		**5451**	**272**		

Tabla 1.3 Programa de ruta según el software *Optrak 4*

Tiene como innovación la representación de la información de cada una de las rutas por medio de histogramas, una práctica que comienza a ser habitual en las interfaces de las aplicaciones informáticas.

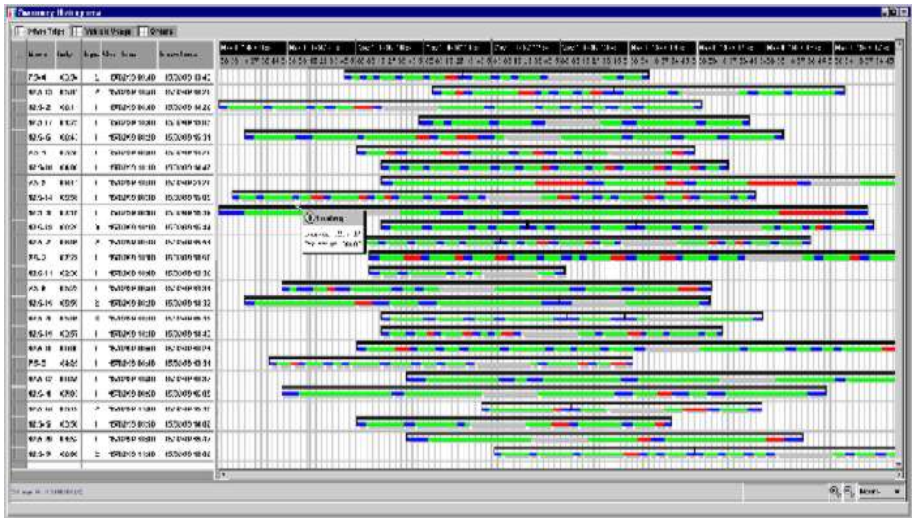


Figura 1.13 Diagramas de Gantt resultantes de la planificación de rutas según *Optrak 4*

Los diferentes colores indican distintos tipos de actividades para cada uno de los conductores. Así se recogen las tareas derivadas de carga, descarga, conducción, esperas, etc. Además se puede utilizar la tecnología de "*arrastre*", para realizar los oportunos cambios en itinerarios y rutas.

Además y como no podía ser de otra forma, la aplicación recoge información geográfica sobre las rutas a realizar, empleando mapas vectoriales, de manera que la visualización de los resultados es mucho más amigable, tal y como se muestra en la figura 1.14. Estos mapas permiten una potencia de visualización de hasta el nivel de las calles.

Finalmente y como diferencia con las aplicaciones anteriores, el software de *Optrak* destaca por incluir un módulo de análisis de sensibilidad, donde se permite almacenar información de las soluciones generadas, para a continuación establecer cualquier tipo de modificación de manera que el usuario puede interactuar con la aplicación conociendo en todo momento la repercusión de los cambios realizados. Como complemento se permite realizar actividades de control sobre las rutas planificadas, introduciendo los datos reales, de manera que se puede utilizar para poder corregir ineficiencias en el futuro y aprender de los errores del pasado. En este sentido se podría considerar como un sistema experto, dada su capacidad de aprendizaje.

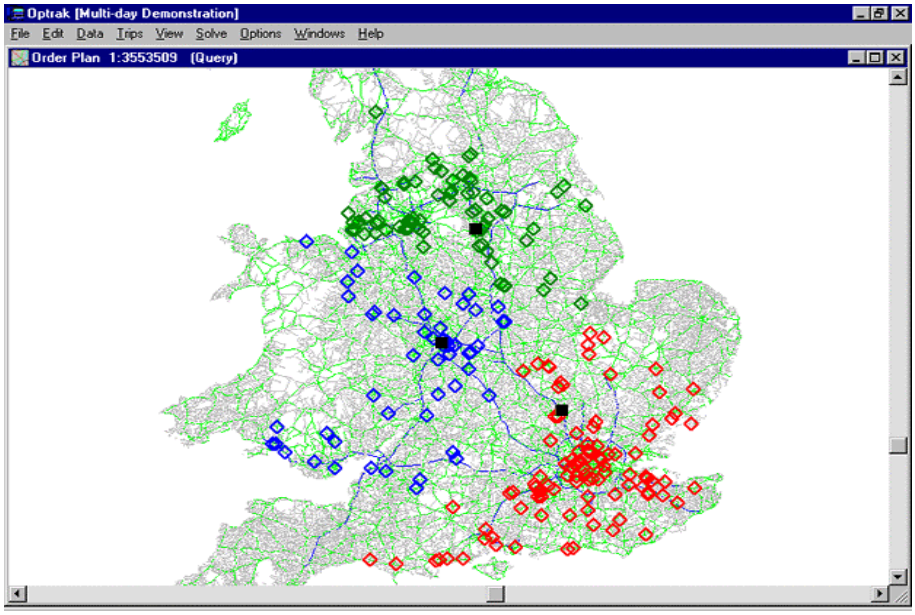


Figura 1.14 Representación sobre mapas vectoriales de las rutas y nodos según *Optrak 4*

Optrak no suministra información técnica sobre los métodos utilizados en su aplicación de manera que se desconoce los algoritmos aplicados. Sí se incluye información relevante a las capacidades de la aplicación para soportar las distintas variantes del problema de planificación de rutas de vehículos[9].

Pedidos	Vehículos y conductores
Envíos y recogidas	Conductores, trailers y tractoras separadas
Transferencias entre depósitos	Horario legal de conductores
División de envíos o agregación	Costes de horas extra
Manejo de códigos postales	Tamaño variable del grupo de conductores
Envíos a domicilio	Multicompartimento
Jerarquía de pedidos	Multitemperatura
Restricciones de acceso de vehículos	Alquiler de vehículos extra

Depósitos	Programación
Múltiples depósitos	Viajes de varios días
Depósitos sin stocks	Horario 24 horas
Disponibilidad de producto en almacenes	Ventanas de Tiempo
Costes de almacenamiento	Rutas fijas
	Diferentes velocidades por vehículo

Tabla 1.4 Características del Software *Optrak 4*

[9] Es interesante visitar la página de *Optrak* (www.optrak.com) para conocer alguna de sus aplicaciones a diferentes empresas del sector.

1.13.4 ArcLogistics Route

Se trata del software diseñado por *ESRI, Environmental Systems Research Institute Inc*. En esta aplicación confluye la experiencia de uno de las empresas punteras en desarrollo de aplicaciones de tipo GIS, con la investigación en sistemas de optimización de rutas. Por ello, el presente software contiene un elevado grado de visualización y sencillez en el manejo, pero en detrimento de las capacidades funcionales del mismo.

Este programa contiene información sobre todo el territorio de Estados Unidos, hasta el nivel de los callejeros, de manera que las distancias y los cálculos realizados llegan a su máximo exponente en cuanto al realismo alcanzado.

La forma de trabajar con esta aplicación es similar a la de los casos anteriores. Es necesario definir el área con la que se pretende trabajar para que el programa utilice los datos de esa región. A continuación se definen los depósitos, almacenes, u otros sitios de interés. Además se establecen los vehículos de los que se dispone, los conductores, los costes por milla y hora de cada vehículo, e incluso alguna característica especial de los vehículos como por ejemplo, si dispone de plataforma elevadora o no.

A continuación se introducen los pedidos para ser optimizados por el programa. Se asignan los pedidos a los camiones y se calcula los tiempos de cada ruta y la secuencia de visitas, de manera que se minimice el tiempo total empleado.

La solución proporciona un mapa con todas las indicaciones de las rutas, e incluso un informe para cada ruta.

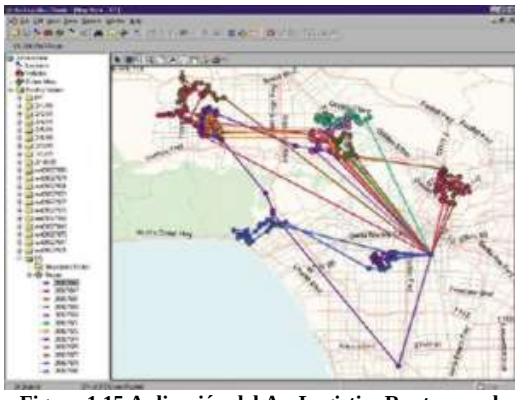


Figura 1.15 Aplicación del ArcLogistics Route para la recogida de basuras en la ciudad de Los Ángeles (USA)

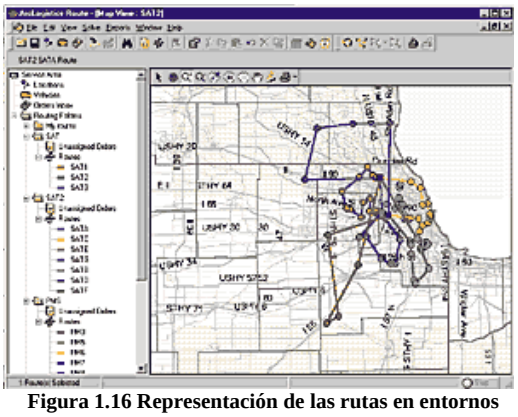


Figura 1.16 Representación de las rutas en entornos urbanos

Como inconvenientes cabe destacar la no diferenciación entre vehículo y conductor, ya que en muchos casos es un factor a tener en cuenta, sobre todo si realiza actividades de tipo comercial, además de las de tipo logístico. Por otro lado, el programa solamente permite establecer cuatro tipos de velocidades medias para los vehículos, en función del tipo de carretera, pero no permite la diferenciación de las mismas en función de la hora

del día de la que se trate, algo que en muchos casos es determinante y que puede hacer fracasar cualquier planificación.

La mayor ventaja de este tipo de software es la inclusión del módulo de geocodificación, consistente en que se puede posicionar en el plano cualquier dirección postal, de manera que no es necesario localizar cada pedido uno a uno de forma manual, sino que el ordenador contiene las bases de datos con la información necesaria para posicionar cada dirección postal de los Estados Unidos.

La propia empresa reporta que los costes generalmente se reducen entorno a un 5 ó 15%, al aplicar este software y que ello permite rentabilizar los 9.000$ de inversión inicial, más los 2.000$ de mantenimiento anual. A mayores de las inversiones necesarias en hardware, dado que se necesita una potencia mínima de procesador de 266Mhz y 1Gb de almacenamiento en disco para los datos geográficos, además de un monitor mínimo de 17 pulgadas para evitar constantes desplazamientos por la pantalla.

Puesto que esta aplicación se ha basado desde sus comienzos en una mayor importancia de la aplicación de los sistemas GIS, cabe destacar que una de sus mayores ventajas es precisamente esa, dado que permite extensiones referentes al seguimiento de los vehículos en tiempo real a través de localizadores GPS y señales de telecomunicaciones de tipo GSM, de manera que es posible tener contacto online con todos y cada uno de los vehículos de la empresa, algo que se ha implantado en algunas de las mayores compañías del sector, como por ejemplo *UPS*

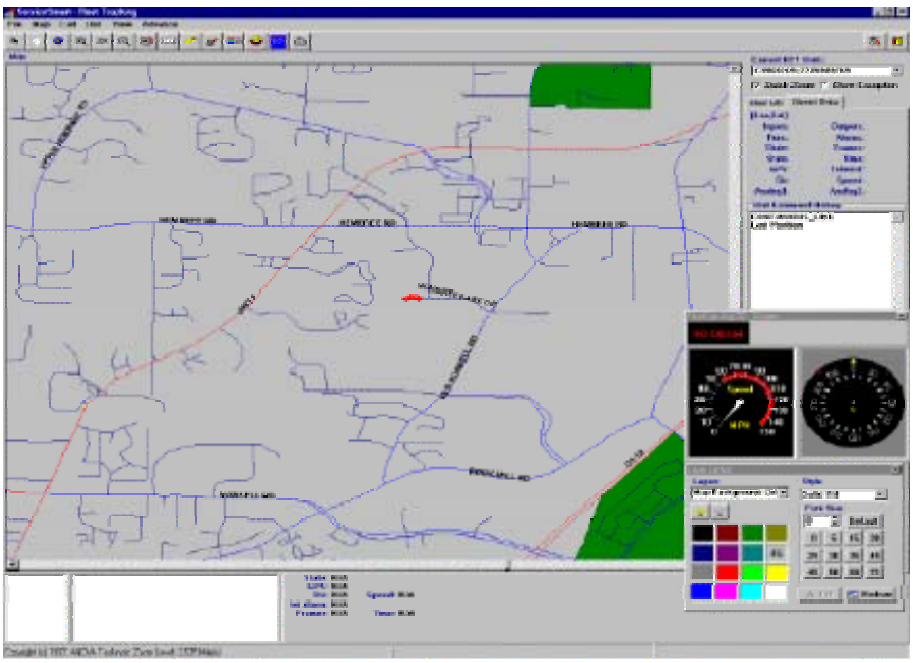


Figura 1.17 Representación del simulador de seguimiento de vehículos

1.13.5 Super-IFMAP de IBM

También destacan los desarrollos de Okano e Hidaka[10], en el centro de investigación de la empresa *IBM*, en Japón. Estos autores han desarrollado principalmente la línea de investigación dedicada a la distribución estratégica de almacenes en el territorio, pero recientemente han diversificado su investigación hacia la planificación de rutas de transporte. La planificación se apoya en los sistemas GIS para calcular las distancias y tiempos de desplazamiento entre los nodos a ser servidos, y de esta manera se efectúa la planificación de las rutas. Posteriormente, una vez que las rutas se han obtenido a través de algoritmos de optimización, éstas se muestran en el mapa digital. El algoritmo utilizado se fundamenta en el problema de agrupación de demandas, y tras algunas simulaciones establecen que son capaces de obtener ahorros cercanos al 15% con respecto a los costes originales Las etapas de planificación descritas anteriormente se corresponden con las siguientes imágenes de la interfaz desarrollada por estos autores, recogidas en la figura 1.18.

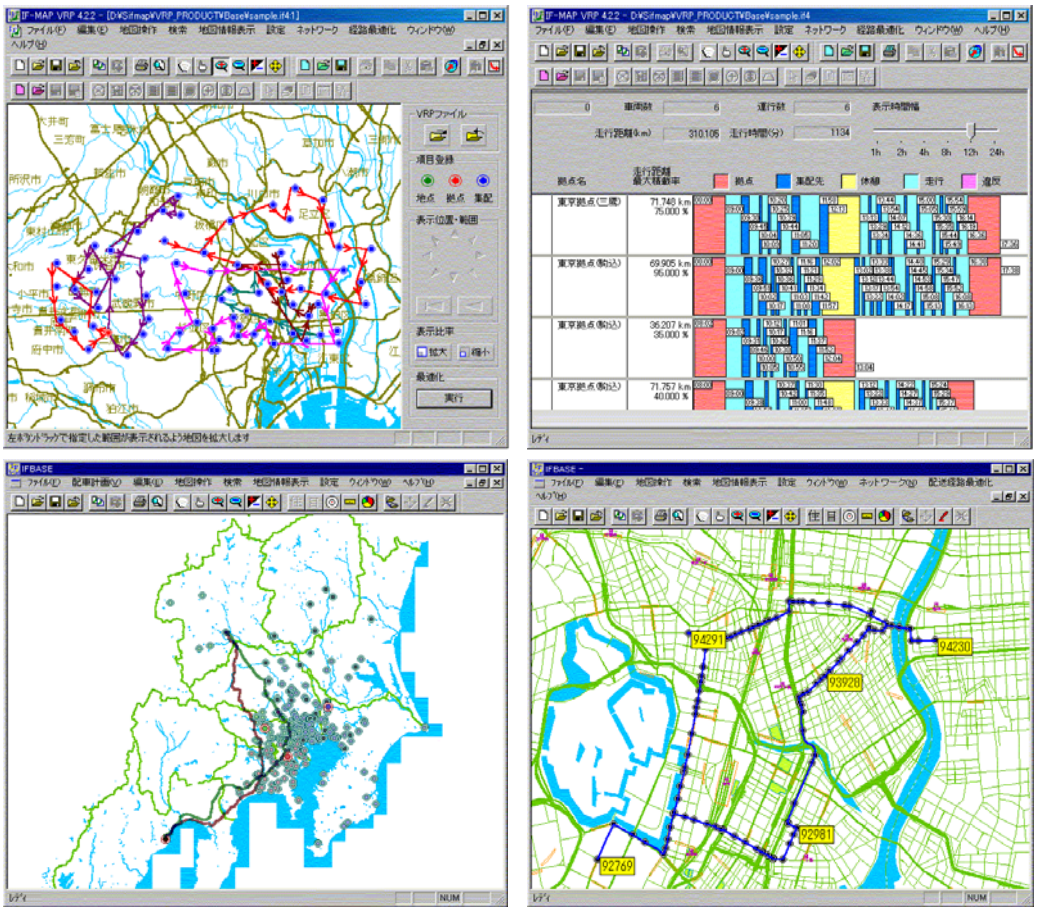


Figura 1.18 Diferentes vistas de la interfaz de *Super-IFMAP*

[10] Véase http://www.trl.ibm.com/projects/optsim/logiopt/index_e.htm

Los sistemas de información geográfica aplicados por estos autores se fundamentan en el *Super-IFMAP*, software de información geográfica de *IBM Japón*. En este sentido se utiliza fundamentalmente para la obtención de las distancias entre todos los nodos a ser servidos.

Una vez resuelto el problema de concentración de las demandas de los clientes, entonces se constituyen las rutas, de manera que la secuencia de visitas de estos nodos clusterizados previamente, se realiza utilizando algoritmos de resolución del TSP con distancias no euclídeas, a través de reglas de cálculo del camino más corto.

1.13.6 Metodología DPS: PriceWaterhouse Coopers

Por último, cabe señalar la creciente preocupación existente en el mundo de la consultoría por lo que también se ha seleccionado el caso de la empresa danesa *DPS*. En este caso, el equipo de investigadores dirigidos por Ole Kessel[11], propone una metodología a la hora de aplicar los Sistemas de Información Logística en las empresas. Tras múltiples experiencias en la implementación de este tipo de proyectos en empresas de distribución, y tras integrarse en la empresa absorbente, establecen un procedimiento para el diseño de los sistemas DSS logísticos. En este procedimiento detallan todas las actividades que han de ser realizadas en la implementación de estos sistemas y que comienzan, al igual que en los casos anteriores, con la geocodificación de los nodos de servicio en un mapa previamente digitalizado. De esta manera es posible obtener todas las distancias y tiempos de desplazamiento entre todos los nodos considerados. De esta manera se obtiene la matriz de distancias para realizar la planificación de las rutas, que de la misma manera se presentan en el mapa. Para estas rutas se aporta toda la información necesaria de la programación, en concreto los momentos de servicio, los tiempos de viaje, los posibles tiempos de espera, los tiempos de descanso, etc. Para ello, se utilizan los diagramas de Gantt que también se proponen en todos los casos anteriores. Las dos imágenes siguientes se corresponden con estas dos fases.

[11] Véase www.imm.dtu.dk/conferences/route2000/

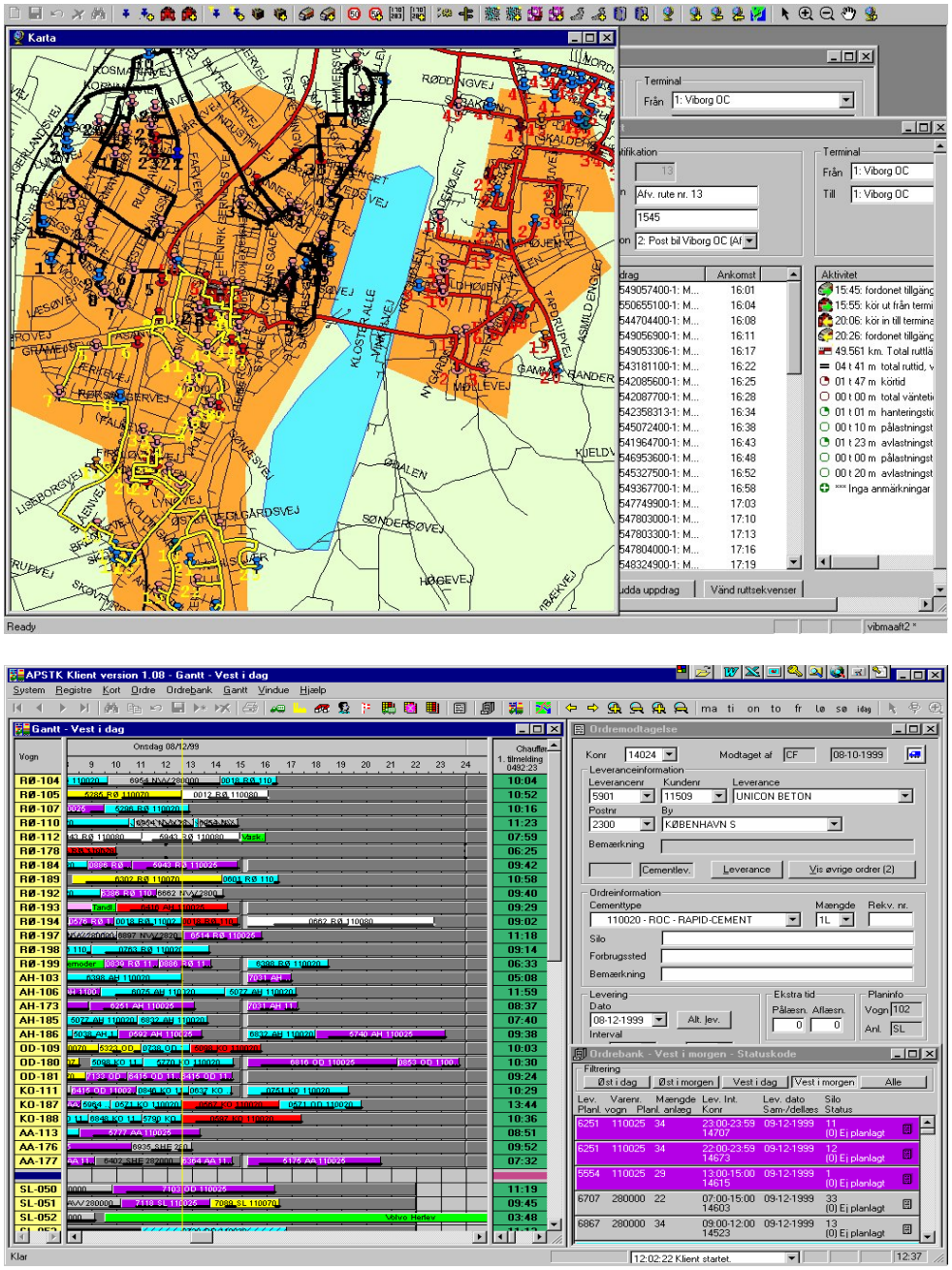
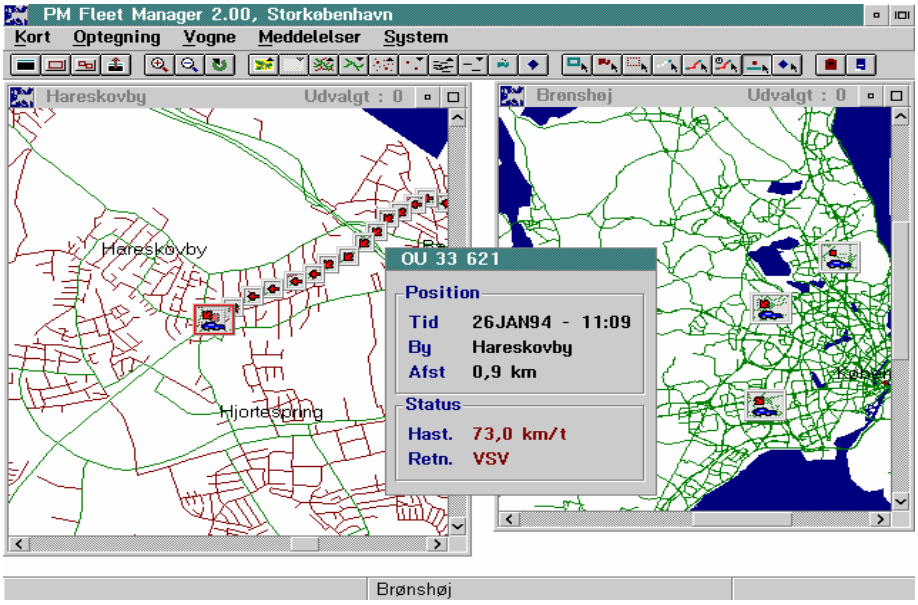


Figura 1.19 Dos vistas de la interfaz desarrollada por la empresa *DPS, (PriceWaterhouseCoopers*) en las que se aprecian la visualización de las rutas sobre un sistema GIS, y los diagramas de Gantt de las rutas.

Ya a continuación, y para facilitar la planificación dinámica de las rutas, así como el establecimiento de posibles rutas alternativas, se proponen los sistemas de seguimiento de flota de vehículos, tanto desde el almacén central, como a partir de una página de *Internet*

corporativa. De esta manera se contemplan todas las actividades derivadas de la búsqueda de la satisfacción total del cliente, que en todos los casos ha de ser la premisa a seguir incurriendo, al mismo tiempo, en los menores costes posibles. Las dos siguientes imágenes se corresponden con estas dos últimas etapas de la planificación.



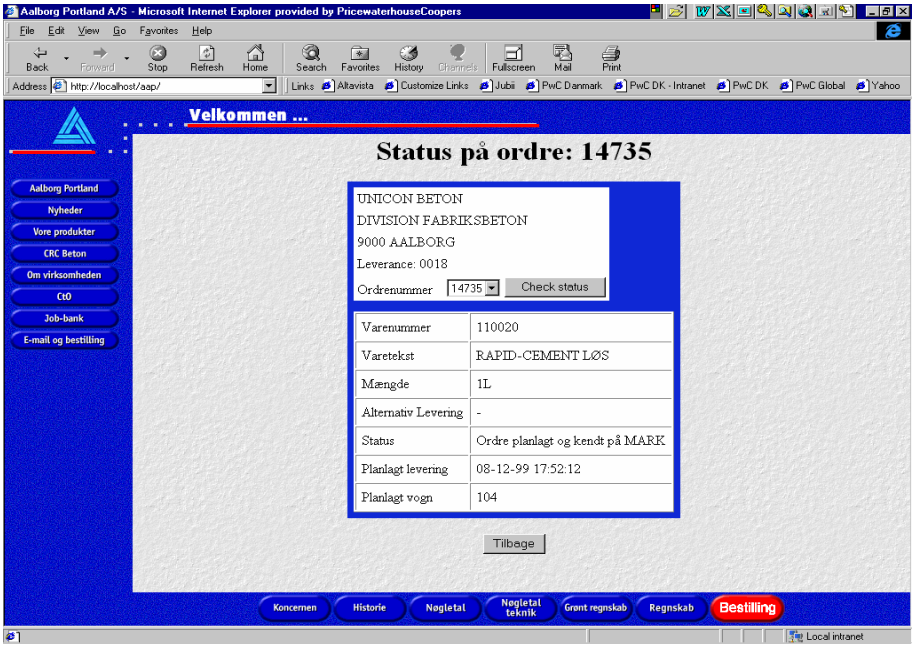


Figura 1.20 Dos vistas del software de *DPS*, correspondientes al geoposicionamiento de los vehículos para la realización del seguimiento, y al seguimiento de envíos por *Internet*.

1.13.7 Análisis de la situación en España

En el caso de España, los desarrollos de los Sistemas de Información Logística para las empresas han quedado relegado a pocas iniciativas, y generalmente de escasa relevancia. Si bien las grandes empresas del sector sí poseen este tipo de programas de planificación, estos derivan de las iniciativas de empresas internacionales. Por ello los desarrollos propios se corresponden con aplicaciones hechas a medida, y generalmente sin la implementación de los sistemas de información geográfica. Sin embargo en los últimos años y motivado por el auge de los sistemas GIS, se empiezan a ver algunas de estos desarrollos. En la revisión bibliográfica se recoge alguna de estas iniciativas a cargo de Adenso et al. (1998) en el caso de la planificación de rutas para el sector lácteo en Asturias. Igualmente Pacheco et al. (2000) desarrollan alguna iniciativa en el campo de la planificación de rutas de autobuses escolares en la provincia de Burgos, a través de la implementación de algoritmos fundamentados en la búsqueda tabú, para el problema conocido como *Pickup and delivery*, donde se consideran las cargas y descargas simultáneas.

En especial, destaca muy recientemente una nueva iniciativa propuesta por la empresa *Geofactory, S.A*.[12] de Santander. Esta empresa nace con la finalidad de aplicar las técnicas de geocodificación de datos para crear y servir mapas digitales de diferentes localidades de España, al igual que las redes de carreteras. En este sentido, y tras una fuerte entrada en las aplicaciones para *Internet*, la empresa se plantea nuevas aplicaciones de este *know-how*, y deciden desarrollar entre otras, la línea de investigación destinada a la planificación de rutas de transporte. Para ello integra diferentes tecnologías, incluidas las derivadas del GPS o de GSM para la localización de elementos móviles. Sin embargo propone una nueva tecnología para el desarrollo de todos los procesos de visualización de los datos. Esta tecnología la denomina *Location Based Systems*, *LBS*.

El sistema integral de *LBS* constituye varias fases entre las que se encuentra la geolocalización, bien a través del *GPS*, o bien a través del *GSM*, la geocodificación, o localización en el territorio de los puntos geolocalizados. La georepresentación, consistente en la visualización de los objetos (en este caso vehículos) en planos digitales, y posteriormente la obtención de rutas de transporte. En este sentido la empresa ha desarrollado ampliamente los sistemas basados en la tecnología GIS, pero a diferencia de estos, los sistemas LBS son fácilmente integrables en entornos de redes informáticas, sobre todo para accesos de múltiples usuarios, de manera que permite su utilización para aplicaciones principalmente pensadas para Internet. En segundo lugar, el LBS constituye una aplicación muy versátil y flexible, por lo que su interfaz es personalizable. Y por último, lo más importante, es que incluye los sistemas de geolocalización integrados, de forma que no es necesario adquirir por la empresa software adicional para realizar la geolocalización. De esta manera aportan tanto los mapas digitales como la geolocalización y geoposiconamiento en los mismos. Además permite la visualización en cualquier tipo de hardware, desde PCs y portátiles, hasta teléfonos móviles, PDAs y TV.

[12] Véase www.geofactory.com/

A través del módulo de *gRouter,* se puede calcular las rutas entre dos o más puntos en una red de carreteras, o bien en ciudad, tanto para realizar a pie, como para vehículos, contemplando la posibilidad de información sobre tráfico en este segundo caso.

Sin embargo, todavía no han incluido las modificaciones necesarias para la aplicación al transporte de mercancías, aunque recientemente han llegado a un acuerdo con la empresa fabricante de hardware *Maxon*, para la implementación e integración de todos los sistemas, con la intención de entrar en el mercado de estas aplicaciones informáticas.

Hasta el momento se ha realizado una revisión de las soluciones existentes para realizar la planificación de rutas de transporte, tanto a nivel de aplicaciones comerciales, como a nivel de proyectos de asesoría ad hoc. Igualmente se ha contemplado la situación en el caso de España, donde si bien la existencia de este tipo de iniciativas ha sido tardía, se augura un futuro prometedor, dado el nivel alcanzado en los desarrollos de los sistemas GIS. En este sentido en los próximos años, nos encontraremos con múltiples actividades encaminadas a la planificación automatizada de rutas de transporte, tanto a nivel de investigación básica, en la búsqueda de nuevos algoritmos, como en la fase de desarrollo de aplicaciones.

2 Conclusión

En este capítulo se ha comentado la especial importancia que tiene y tendrá en los últimos años la evolución en los Sistemas de Información Logística o SIL, para los tres niveles de planificación empresarial, estratégico, táctico y operativo. Las circunstancias actuales en los mercados apuntan a que cada vez más el desarrollo de sistemas de planificación de rutas de transporte eficientes y eficaces, reportará fuertes ventajas competitivas a las empresas, sobre todo en cuanto a la reducción de costes se refiere. Por ello, y ante una creciente tendencia en el aumento de los precios de los carburantes, es manifiesta la necesidad de obtener métodos que permitan la programación de las visitas a realizar por las diferentes rutas en las que opera la empresa, incurriendo en los menores costes posibles.

Igualmente se hace necesario el establecimiento de sistemas de control de las planificaciones y programaciones previamente realizadas, sobre todo para la corrección de estas programaciones ante cualquier posible contingencia o imprevisto. La implementación de estos sistemas de seguimiento o control se hacen especialmente deseables en el caso donde la planificación que realizan las empresas es de tipo dinámico.

Actualmente la forma de planificación existente en las grandes empresas ya se fundamenta en potentes herramientas informáticas que centralizan esta actividad arrojando soluciones a la planificación de rutas de transporte, así como permitiendo el seguimiento de las mismas, incluso en tiempo real. Estas soluciones requieren actualmente de fortísimas inversiones tanto en software como en hardware, y por ello solamente son rentables en las grandes corporaciones.

Por otro lado las PYMES se han de conformar con perseguir líneas de planificación de rutas mucho más intuitivas, fundamentadas en las experiencias pasadas del planificador, así como en la base de su conocimiento tácito. Generalmente este tipo de planificación arroja buenos resultados, aunque mejorables a través de la implementación de técnicas matemáticas de optimización. Para ello se requiere el desarrollo de modelos de optimización y métodos heurísticos que faciliten al planificador la toma de estas decisiones, sin que para ello tenga que afrontar fortísimas inversiones en infraestructuras.

En este marco se plantea el presente trabajo cuyo objetivo último es el análisis de las decisiones sobre la planificación de rutas de transporte, y más concretamente sobre las decisiones relativas a la construcción de las sucesivas visitas a clientes, y su integración en rutas reales de reparto o recogida. De esta manera, y tras un estudio detallado de los métodos de construcción de rutas de transporte existentes en la literatura, y que se desarrolla en el capítulo 2, se desarrolla un método nuevo de construcción de rutas, con la intención de analizar el porqué de las decisiones de adición de nodos a las rutas, así como los problemas e inconvenientes de estas reglas de decisión, con la única finalidad de comprender y aprender sobre estas reglas para su futura implementación en Sistemas de Apoyo a la Toma de Decisiones en PYMES con problemas de tipo logístico. En el

capítulo 3, se desarrolla el método de construcción de rutas resultado de la investigación realizada, y para el que también se ha diseñado una interfaz de comunicación entre el algoritmo y el usuario. Finalmente se ha validado la eficacia del algoritmo sobre un conjunto de problemas tipo que se recogen en la literatura, y cuyos resultados se comentan en el capítulo 4 de este trabajo.

Por último se discute la aplicabilidad de los resultados de esta investigación a la problemática real de las empresas, destacando las ventajas e inconvenientes del método desarrollado en el curso de la presente investigación, así como detectando posibles áreas de implementación del mismo como base para la elaboración de *DSS* logísticos.

Capítulo 2

Estado de la cuestión

El objetivo principal de este capítulo es mostrar los diferentes planteamientos contemplados por la literatura hasta la actualidad para resolver los problemas generales de un sistema logístico de rutas de reparto en la empresa. Para abordar este desarrollo se ha dividido el capítulo en cuatro secciones claramente diferenciadas. En primer lugar y tras una breve introducción al tratamiento del problema por la literatura, se aborda el problema del viajante, también conocido como el *Traveling Salesman Problem* o TSP, como punto de partida de la optimización de rutas, así como sus posibles soluciones atendiendo a los algoritmos más conocidos. En la segunda sección del capítulo se aborda de manera más extensa el problema de planificación de rutas, denominado *Vehicle Routing Problem* o VRP, con sus múltiples variaciones y los diferentes modelos desarrollados por los investigadores más conocidos en esta área. En tercer lugar se centra la atención en el problema de planificación de rutas de transporte contemplando la existencia de plazos de entrega o ventanas de tiempo, más conocido como VRP con *Time Windows* o VRPTW, y que servirá de base para establecer el modelo desarrollado en el presente trabajo como soporte a la planificación de rutas para empresas de transporte. En este sentido, aunque el objeto del presente trabajo sea la investigación en el problema concreto del VRPTW, ha sido necesario recopilar toda la información relativa a los problemas del TSP y del VRP, dado que la mayor parte de los trabajos en la línea del VRPTW se fundamentan en los algoritmos y modelos recogidos en las dos áreas de investigación anteriores. Finalmente concluye el capítulo con una recopilación de casos reales dónde se describen experiencias de aplicación de las técnicas de optimización de rutas a empresas u organizaciones donde existe este tipo de problemas.

2 Introducción

De acuerdo con Eilon, Watson-Gandy y Christofides (1971), la logística se podría definir como "*el suministro de bienes y servicios desde un punto de oferta a otro punto de demanda*". De esta manera, un sistema logístico cubriría el proceso completo del movimiento de materias primas y otros *inputs* desde los proveedores hasta las plantas de transformación, el movimiento de los productos hasta los diversos depósitos o almacenes y el envío de los productos terminados a los clientes finales. Otros diferencian entre las actividades de la empresa relacionadas con la logística interna y aquellas relacionadas con la logística externa. Las actividades de logística interna consistirían en la gestión de los productos terminados y en curso de fabricación dentro de la empresa así como todas las actividades relacionadas con la gestión de almacenes. Las actividades de logística externa supondrían todas aquellas operaciones desde la salida de las mercancías de la fabrica o almacén hasta su suministro a los clientes o consumidores finales. Es en este campo donde se centra la atención del presente trabajo, más que en la propia gestión de almacenes.

De manera más concreta, las actividades de logística externa de una empresa incluyen todos los movimientos de los productos desde la planta de fabricación hasta el cliente final, incluyendo su transporte. El envío final es el componente más costoso de toda la cadena de distribución y de ahí el énfasis tan importante a la hora de planificar una red de rutas lo más eficientemente posible y manteniendo siempre un nivel de eficacia mínimo, garantizando la satisfacción de los clientes.

Los sistemas de planificación de rutas engloban una serie de decisiones que se pueden agrupar en tres grandes áreas: "estratégicas, tácticas y operativas", de acuerdo con el planteamiento de Bodin et al. (1983). Las primeras, se corresponden con las decisiones relativas a la localización de las instalaciones (plantas, almacenes, depósitos). En cuanto al nivel táctico, se correspondería con la determinación de la mezcla óptima en la flota de vehículos y el propio número de vehículos a utilizar. Finalmente las decisiones más operativas se corresponden con la propia planificación de rutas y de horarios de servicio, así como la asignación de personal a cada una de las rutas de reparto. Otros autores (Adenso et al., 1998) reconocen diferentes jerarquías en el planteamiento de los objetivos del Sistema de Información Logística, derivando en múltiples subproblemas que han de resolver de manera escalonada.

Todas estas decisiones están íntimamente relacionadas y en muchos casos es difícil diferenciar estos tres niveles de decisión, por lo que no deben ser interpretados de manera estricta. Además de este tipo de decisiones, debemos de considerar las siguientes:

- Determinación del área servida por cada depósito
- Determinación de la flota y de la diversidad de vehículos disponibles en cada depósito

- Reparto de las actividades de transporte entre la red de la empresa y transportistas externos.
- Nivel de satisfacción del cliente deseado: frecuencia de envíos al cliente, y puntualidad de los mismos.

Una vez tomadas las decisiones sobre las variables consideradas, la empresa tratará de minimizar los costes de toda la distribución física, mediante el establecimiento de rutas y horarios, que se corresponden respectivamente con los problemas de VRP y *Scheduling*, y que se refunden en el problema del VRPTW. Esta optimización ha de hacerse por medio de un algoritmo matemático que establezca la distribución de las rutas y los horarios, algo que dependerá, en última instancia, de las variables mencionadas anteriormente.

En este trabajo se ha puesto un énfasis especial en la programación de rutas como proceso operativo, teniendo en cuenta que la planificación estratégica correspondiente a la situación de los almacenes y composición de la flota de vehículos son decisiones a largo plazo y que por ello se realizan mediante otro tipo de criterios. Es posible asumir que el objetivo último de los sistemas de planificación de rutas es la minimización de costes. Sin embargo, pueden existir otros objetivos secundarios tales como la seguridad del transporte, la minimización del impacto medioambiental, o la fluidez del tráfico (en especial en los sistemas públicos de transporte).

La literatura revisada para desarrollar este trabajo se puede dividir en tres grandes grupos:

- En primer lugar nos encontramos con aquellos trabajos basados en la descripción de casos reales de planificación de rutas de transporte. Se trata en general de descripciones de casos en los cuales se presenta la aplicación de Sistemas de Información Logística en diferentes empresas, tratando siempre de exponer las dificultades de cada caso concreto y cómo los autores han resuelto dichos escollos a la hora de implantar un sistema de gestión de rutas.
- En segundo lugar, se puede identificar otro gran grupo de trabajos que describen de forma más específica los modelos matemáticos y algorítmicos de solución del problema básico de establecimiento de rutas y que sirven de base a los primeros, en concreto los problemas del TSP y del VRP. Por ello es necesario comenzar por una primera aproximación a estos modelos.
- Y ya por último la línea de investigación relativa al problema del VRPTW, que se fundamenta en el problema básico el VRP, pero en el que se incluyen restricciones temporales tanto en los servicios a los clientes en cuestión, como en los momentos de apertura y cierre del depósito central desde donde parten todos los vehículos. En este sentido se presenta como el tipo de problemas más cercanos a la problemática empresarial, dado que cada vez más los clientes de las empresas de transportes exigen disponer de sus mercancías en determinados momentos temporales, sobre todo en aquellos casos donde mayormente se aplican las técnicas conocidas del *Just in Time*. Igualmente las hipótesis de ventanas de tiempo para el depósito central, suele hacer referencia a la duración de la jornada laboral de los conductores de los vehículos, por lo que es necesario

que tengan un momento inicial de partida, y un límite temporal de llegada a dicho depósito. En este sentido se trata de los problemas más realistas en cuanto a la planificación de rutas de transporte para el reparto de mercancías, o en el caso de cargas negativas, el problema de recogida de mercancías.

El problema básico del VRP se centra en un conjunto de nodos o arcos que han de ser servidos por una flota de vehículos. No existen restricciones en cuanto al orden en el cual han de ser visitados estos centros. En este sentido, y a diferencia del VRPTW, se trata de un problema de optimización geográfica, donde se busca el menor número de rutas con distancia total mínima, sin mayores restricciones que las derivadas de la capacidad de los vehículos que cubren esas rutas.

Una ruta es una secuencia de nodos que un vehículo ha de visitar, así como las unidades de servicio que ha de suministrar (kilogramos, unidades, litros, tiempo, etc.). En principio, el problema del VRP es un problema espacial. Sin embargo, este planteamiento inicial puede incorporar múltiples variaciones tratando siempre de aproximarlo al mundo real. Algunas de estas variaciones se recogen en la tabla 2.1.

VRP (Vehicle Routing Problem). Se trata del enfoque genérico de resolución de rutas en base al problema espacial, es decir, la configuración de diferentes rutas compuestas por visitas secuenciales para cada vehículo.
VRPTW (VRP con Time Windows). Consiste en el desarrollo del planteamiento general del VRP, pero teniendo en cuenta el tiempo. En este caso se añade la restricción consistente en que cada una de las visitas solamente se puede realizar en un determinado intervalo de tiempo, de manera que el vehículo ha de estar en ese nodo en dicho intervalo para poder realizar el servicio.
CVRP (Capacited VRP). Se trata del problema general pero añadiendo la restricción de capacidad de cada uno de los vehículos. En este sentido y a la hora de realizar los repartos, es necesario tener en cuenta alguna medida de capacidad del vehículo, como por ejemplo volumen o peso y la misma medida a la hora de valorar los servicios realizados, de manera que cada vehículo tiene su capacidad restringida.
Inventory VRP. Se trata del problema del VRP teniendo en cuenta además el establecimiento de políticas de gestión de almacenes en base a las cuáles es necesario establecer una cantidad de pedido óptima para ser servida de cada vez, así como el momento óptimo de servicio, de manera que los clientes no queden desaprovisionados ni tampoco tengan excesivas cantidades de producto almacenadas.
VRP with Backhauls. Se trata del establecimiento del problema básico, pero teniendo en cuenta que existen clientes que han de ser visitados en el camino de ida del vehículo y otros que han de ser visitados a la vuelta. Este tipo de problemas se puede asociar a aquellos en los que existen recogidas y repartos, o repartos y recogidas, es decir, donde los servicios han de seguir un cierto orden preestablecido.
VRP estocástico. Se trata de la consideración del problema del VRP teniendo en cuenta que las demandas son aleatorias. En el VRP general las demandas son deterministas y conocidas de antemano, mientras que en el VRP estocástico se trata de demandas desconocidas que siguen una determinada distribución probabilística.

Tabla 2.1 Clasificación de los problemas derivados del VRP

El propósito de este trabajo es el diseño de un modelo que permita la planificación de los envíos de una empresa a múltiples puntos de servicio, en los momentos en los que se requiere dicho servicio por parte de los clientes. En este sentido se aborda directamente el problema del VRPTW, a través del análisis de las decisiones empresariales en el caso de la programación de estas rutas de transporte. Para ello se estudian detalladamente este tipo de decisiones empresariales, tratando de integrar todas ellas en un modelo matemático que sea aplicable a cualquier tipo de problemática empresarial en la línea del VRPTW. Para ello se desarrolla, en el capítulo 3, el modelo propuesto por el autor y se recogen en el capítulo 4 los resultados de aplicación de este modelo.

Para desarrollar este modelo de gestión de rutas de transporte es necesario comenzar por comprender el planteamiento básico del Problema del Viajante (TSP), una simplificación del problema del VRP, en la cuál se considera una única ruta.

3 TSP: Traveling Salesman Problem

El planteamiento básico descrito en las líneas precedentes tiene su origen en el problema del viajante, más conocido como TSP (del inglés *Traveling Salesman Problem*). El problema se plantea en términos de establecer cuál es la ruta mínima entre todos los puntos de servicio o nodos, de manera que la ruta pase una sola vez por cada nodo.

Existen múltiples variantes del TSP, pero quizás la más común es considerar las distancias como simétricas, en contra de otros modelos o planteamientos donde se consideran las distancias asimétricas. Aunque generalmente se asume la primera hipótesis por resultar suficientemente realista, existen otros casos en los que sí es necesario considerar costes o distancias asimétricas, sobre todo en planificación de cortas distancias, como por ejemplo en ciudades, debido a la existencia de carreteras de un único sentido y direcciones prohibidas. En este caso el tipo de problema se modeliza a través de la optimización basada en redes, y que se centran en el Problema del Cartero Chino, *Chinese Postman Problem* o CCP.

En términos formales el problema del TSP se plantea en los siguientes términos:

Sea una red $G = [N, A, C]$ definida por un conjunto N de nodos, A un conjunto de arcos y $C = [c_{ij}]$, una matriz donde c_{ij} representa la distancia o coste de desplazarse del nodo i al nodo j. La solución del problema se plantea a través del establecimiento de un ciclo Hamiltoniano[13] (es decir, que pase por todos los nodos una sola vez), que minimice la distancia recorrida.

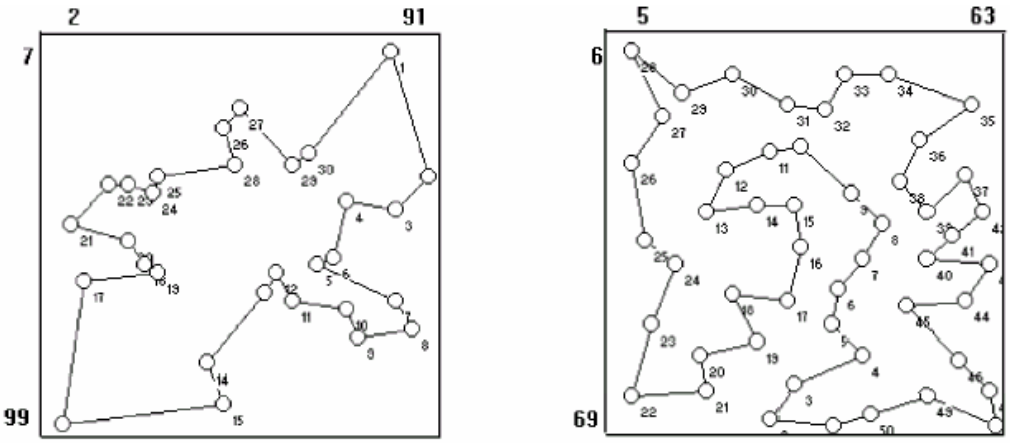


Figura 2.1 Resolución de 2 TSPs, Oliver30 y Eilon50, utilizando el algoritmo de Myagkih, Savelev y Kureichik.

El problema del TSP es un problema de tipo NP Complete (Karp, 1972), es decir que hasta la fecha no se ha encontrado ningún método exacto de cálculo cuyos tiempos de

[13] Para una mayor comprensión de la teoría de grafos es conveniente revisar Berge (1970)

computación evolucionen de forma polinómica con respecto al número de nodos. Por contra, todos los métodos exactos desarrollados hasta la fecha implican tiempos de computación que varían de forma exponencial con respecto al número de nodos, por lo que la resolución de este problema a través de estos métodos se presenta ineficiente para problemas con un elevado número de nodos, requiriendo amplios períodos de computación que en ocasiones se extienden durante meses para la resolución de problemas de cientos de miles de nodos.

Consecuentemente se han desarrollado múltiples modos de resolver el problema, empleando algoritmos heurísticos o de aproximación que permiten calcular soluciones cercanas al óptimo en tiempos mucho más reducidos, de manera que su utilización, a pesar de no ser tan eficaz como los algoritmos exactos, sí resultan enormemente eficientes ya que la calidad de los resultados comparado con el esfuerzo en tiempo de computación es excelente, consiguiendo soluciones en torno al 1% superiores a los costes o distancia del límite inferior de la solución óptima.

Es interesante la comparación de los algoritmos heurísticos de resolución de TSPs, para establecer cuál ha sido la evolución de los mismos, lo que repercute directamente en el nacimiento de nuevos modelos para la resolución de los VRP, en gran medida basados en los planteamientos de los primeros.

Para establecer la evolución de los planteamientos de resolución de TSPs se proporcionan evaluaciones de cada algoritmo mediante su aplicación a problemas tipo desarrollados en la literatura, de manera que el lector se puede forjar una idea de las mejoras conseguidas y de la bondad de cada método. La fuente de problemas tipo más conocida y utilizada por la literatura es la biblioteca de casos de Reinelt (1992), denominada TSPLIB[14]. Esta biblioteca tiene casos de hasta 85.900 nodos, generalmente provenientes de la fabricación de tarjetas de circuitos integrados (Korte, 1988) y cristalografía con rayos X (Bland y Shallcross, 1989), o incluso de casos de distribución geográfica de ciudades, como el usa13509, cuya solución óptima se presentó en el capítulo 1.

Además de esta biblioteca, existen otras formas o fuentes de problemas tipo, desarrollados de manera aleatoria por algunos autores como Johnson et al. (1996). Estos investigadores crean de manera aleatoria problemas de hasta 10^6 nodos.

2.3 Soluciones óptimas del TSP

La solución óptima del TSP requiere la utilización de la programación matemática, que se ha de resolver a través de la programación lineal entera, generalmente a través del método Simplex, o bien a través de programas informáticos específicos como el *CPLEX*, diseñado *ad hoc* para la resolución de problemas de programación matemática.

Es interesante determinar de manera formal las expresiones del TSP para entender el modelo. Partimos de la hipótesis de que las distancias son simétricas y por ello los costes.

[14] Disponible a través de ftp anónimo en www.softlib.rice.edu

Por tanto $c_{ij} = c_{ji}$. El objetivo entonces es construir un ciclo Hamiltoniano o ruta que pase por todos los nodos y que minimice la distancia total recorrida.

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Supongamos que $x_{ij} = \{1, 0\}$ según el arco ij esté o no en el ciclo. La solución consiste entonces en la obtención de la matriz $X = (x_{ij})$. Sin embargo, para que estos arcos formen un ciclo continuo o ciclo Hamiltoniano, y no subciclos, es necesario establecer una serie de restricciones adicionales que se expresa a través de las siguientes formulaciones:

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall j \qquad \text{Garantiza una única entrada a cada nodo}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall i \qquad \text{Garantiza una única salida de cada nodo}$$

$$x_{ij} = 0 \text{ ó } 1 \qquad i, j = 1, \ldots, \text{n}$$

$$x_{ij} \in S \qquad \text{Evita que se produzcan subrutas, donde}$$

$$S = \left\{ x_{ij} : \sum_{i \in Q} \sum_{j \notin Q} x_{ij} \geq 1 \quad \text{para cualquier subconjunto no vacío } Q \text{ de } N \right\}$$

Este es el planteamiento formal del modelo cuya resolución inmediata se obtendría a través de la programación matemática, algo que, como se comentó inicialmente, sólo es eficiente para problemas de reducidas dimensiones.

A pesar de ello, se han desarrollado en la literatura algunas formas de obtener lo que se han denominado los límites inferiores del valor óptimo del TSP. En este sentido, si bien no se calcula el valor de la solución óptima, sí es posible calcular el límite inferior de esta solución óptima. Entre estos planteamientos destacan los trabajos de Held y Karp (1970 y 1971).

2.3.1 El límite inferior de Held y Karp

Estos autores presentan un modelo de aproximación al óptimo mediante la obtención de un límite inferior cercano al óptimo. Este límite no es una solución al problema, aunque sirve como valor mínimo de referencia aplicable a cualquier método. Generalmente el valor óptimo de la solución se halla entre un 1% y un 2% sobre este límite inferior.

Este límite inferior propuesto por Held y Karp (1970 y 1971), se puede calcular a través de su formulación, mientras que el valor óptimo seguirá siendo desconocido. La formulación inicial para el cálculo del límite inferior es igual que en el problema original, aunque en este caso las dos primeras restricciones que garantizaban una entrada y una salida en cada nodo, se han sustituido por dos nuevas restricciones que garantizan un grado dos en todos los nodos, y que son equivalentes a las del problema inicial:

$$\min_x \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Sujeto a

$$\sum_j x_{1j} = 2$$

$$\sum_j x_{ij} + \sum_k x_{ki} = 2 \qquad \text{i} = 1,2,..., \text{n}$$

$$\sum_i \sum_j x_{ij} = n$$

$$X = \left(x_{ij}\right) \in S$$

$$S = \left\{ x_{ij} : \quad \sum_{i \in Q} \sum_{j \notin Q} x_{ij} \geq 1 \quad \text{para cualquier subconjunto no vacío } Q \text{ de } N \right\}$$

$$x_{ij} = 0 \text{ ó } 1 \qquad i, j, k = (1, \ldots, n)$$

A este problema le denominamos problema *P*.

De esta manera se garantiza que cada nodo tiene grado dos a través de las dos primeras restricciones y que además el número total de arcos en la solución será de *n*, es decir, tantos arcos como nodos existentes, ya que *n=N*. Entonces se puede calcular un límite inferior si se resuelve la relajación lagrangiana a través de la inclusión de la segunda restricción en la función objetivo:

$$\min_x \sum_i \sum_j c_{ij} x_{ij} + \sum_i \lambda_i \left\{ \sum_j x_{ij} + \sum_k x_{ki} - 2 \right\}$$

Donde $\lambda_i = (\lambda_1, \lambda_2, ..., \lambda_n)$ es el vector de los multiplicadores de Lagrange. Si ahora se realizan algunas operaciones en la función objetivo, entonces tenemos:

$$\min_x \sum_i \sum_j (c_{ij} + \lambda_i + \lambda_j) x_{ij} - 2\sum_i \lambda_i$$

Sujeto a

$$\sum_j x_{1j} = 2$$

$$\sum_i \sum_j x_{ij} = n$$

$$X = \left(x_{ij}\right) \in S$$

$$S = \left\{ x_{ij} : \quad \sum_{i \in Q} \sum_{j \notin Q} x_{ij} \geq 1 \quad \text{para cualquier subconjunto no vacío } Q \text{ de } N \right\}$$

$$x_{ij} = 0 \text{ ó } 1 \qquad i, j = (1, \ldots, n)$$

A este problema lo denominamos LR_λ

Sean z y $z_{D(\lambda)}$ los valores óptimos de los problemas P y LR_λ respectivamente. Entonces si x_{ij}^* fuese la solución óptima para P, entonces se cumpliría que:

$$z_D(\lambda) \leq \sum_i \sum_j c_{ij} x^*_{ij} + \sum_i \lambda_i \left(\sum_j x^*_{ij} + \sum_k x^*_{ki} - 2 \right) = z$$

Por lo tanto, LR_λ proporciona una familia infinita de límites inferiores para la longitud de la ruta óptima, uno para cada valor de λ. De esta manera para cada valor de λ, el valor de $z_{D(\lambda)}$, es fácil de determinar, de forma que el mejor de estos límites inferiores vendrá dado por la siguiente expresión:

$$\max_\lambda z_D(\lambda)$$

La ventaja de este método es que la solución de LR_λ es mucho más sencilla que la del problema original *P*. Para los casos donde el número de nodos no es excesivo, se puede obtener de manera exacta a través de la programación lineal, utilizando el Simplex, lo cuál permite obtener el valor exacto para problemas de hasta 30.000 nodos. De esta manera se obtiene el límite inferior del valor óptimo, que por término medio supone un 2% menos que el valor del óptimo real, tal y como se refleja en la figura 2.2

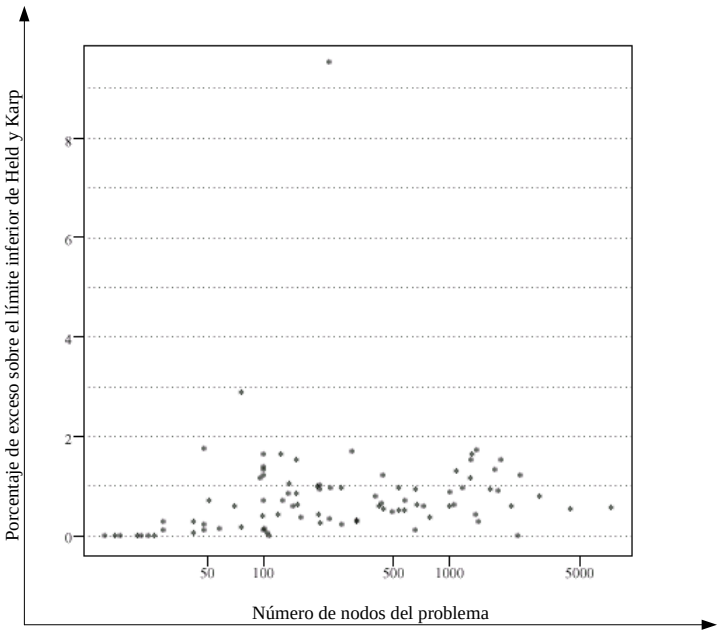


Figura 2.2 Comparación en porcentaje de exceso del valor óptimo sobre el límite inferior propuesto por Held y Karp (1970 y 1971), para algunos problemas del TSPLIB, en función del número de nodos. Johnson et al. (1996)

Sin embargo, para problemas de tamaño superior y hasta un total de hasta un millón de nodos, la resolución del límite inferior se ha de obtener a través de una aproximación por iteración de las secuencias de problemas lineales restringidos, cada uno incluyendo solamente un conjunto de restricciones, utilizando una subrutina para identificar las restricciones que están siendo violadas. Esta es la rutina propuesta también por Held y Karp (1970 y 1971) en sus trabajos.

Esta aproximación se ha mejorado a través de multitud de variantes de las rutinas de comprobación. Algunas de estas rutinas las desarrollan autores como Helbig-Hansen y Krarup, (1974); Held, Wolfe y Crowder, (1974) o Johnson, McGeoch, y Rothberg, (1996), quienes consiquen calcular el límite inferior de manera exacta para problemas de hasta 33.820 ciudades y de hasta 1 millón de manera aproximada. Las aproximaciones suelen ser bastante buenas dado que generalmente se obtienen diferencias de un promedio de 0.01% del valor verdadero del límite inferior.

El problema básico de las soluciones óptimas es el tiempo necesario para resolver el planteamiento inicial, por lo que para problemas que contemplen un elevado número de nodos, o de restricciones, es necesario desarrollar algoritmos heurísticos o de aproximación, que de alguna manera se acerquen lo más posible a la solución óptima del problema. Estos algoritmos son considerados en la siguiente sección.

2.4 Aproximaciones heurísticas al TSP

Podemos clasificar este tipo de enfoques en tres grandes grupos:

- **Procedimientos de construcción**: Tratan de construir rutas partiendo del planteamiento inicial de manera que la solución se aproxime al óptimo.
- **Procedimiento de mejora**: Parten de una solución determinada y tratan de mejorarla.
- **Procedimientos compuestos**: Combinan ambas técnicas para la construcción de la solución final.

Comenzamos la descripción de los métodos de construcción de rutas, considerados como los algoritmos de primera generación y caracterizados por su rapidez y su sencillez. Se trata generalmente de algoritmos enormemente intuitivos y de fácil comprensión.

2.4.1 Algoritmos de construcción de rutas

En este apartado se describen y analizan los métodos heurísticos más sencillos para la construcción de rutas consistentes en la utilización de mecanismos de unión o inserción. Estos algoritmos en general gozan de las ventajas derivadas de su rapidez en el cálculo de la solución. Sin embargo suelen proporcionar soluciones mediocres, que suelen distar del óptimo entre un 20% y un 5%. Se trata de los primeros algoritmos desarrollados por los diferentes investigadores y su mención en este trabajo no obedece a que supongan novedades en el estado del arte, pero sí es necesario dado que muchos de los actuales algoritmos se basan en estos primeros enfoques para proporcionar nuevos resultados. En las siguientes líneas se comentan algunos de estos algoritmos.

2.4.1.1 Procedimiento del punto más cercano (Nearest Neighbor)

Se trata de un método muy simplista consistente en la elección del nodo más cercano para ir trazando los arcos de unión y de esta manera construir la ruta final. Las fases o etapas de este método son las siguientes:

1. Elegir cualquier nodo como comienzo de la ruta.
2. Encontrar el nodo más cercano al último nodo añadido a la ruta y añadirlo a la misma.
3. Repetir el segundo paso hasta completar todos los nodos y finalmente unir los nodos primero y último.

Se trata del proceso más sencillo y simplista, aunque resulta de enorme aplicación práctica por su sencillez. Los resultados obtenidos a través de este método son

generalmente muy pobres y muy fácilmente mejorables, de ahí que su aplicación requiera utilizar en un segundo momento procedimientos de mejora de rutas. Una de las ventajas del método es que se trata de un método completamente determinista, dado que siempre proporciona la misma solución.

3.1.1.1 Método de los ahorros

Este método, desarrollado inicialmente por Clarke y Wright (1964), consiste en trazar la ruta que maximice los ahorros de transporte en cada arco. Para ello el método persigue las siguientes fases:

1. Elegir un nodo cualquiera como nodo inicial, el cuál denotaremos como *1* y suponer las visitas a cada uno de los nodos restantes como de ida y vuelta al nodo inicial.
2. Calcular los ahorros $s_{ij} = c_{1i} + c_{1j} - c_{ij}$ para todo *(i, j)*, si estos se uniesen para una vista secuencial de ambos. Es decir, se contemplan los ahorros generados al unir dos puntos, en vez de realizar dos visitas independientes, de manera que el ahorro conseguido será el viaje de vuelta del primero, más el viaje de ida al segundo, menos el coste del viaje entre ambos.
3. Ordenar los ahorros de mayor a menor.
4. Comenzando por el primero de la lista ir formando subrutas uniendo los nodos *i* y *j* adecuados. Repetir dicha operación hasta completar la ruta.

3.1.1.2 Procedimientos de inserción

Los procedimientos de inserción tienen en cuenta una subruta con *k* nodos en la iteración *k* para determinar qué nodo debería incluirse a continuación y en qué lugar o posición de la subruta considerada.

A: Inserción más Cercana.

1. Comenzar con un subgrafo consistente únicamente en el nodo *i*.
2. Encontrar el nodo *k* tal que c_{ik} es mínimo y formar la subruta *i-k-i*.
3. Dada la subruta anterior, buscar el nodo *k* más cercano a cualquier nodo que ya esté en la ruta.
4. Buscar el arco *(i, j)* en la subruta que minimiza $c_{ik}+c_{kj}-c_{ij}$. Insertar *k* entre *i* y *j*.
5. Repetir el paso 3 hasta completar la ruta.

B: Inserción más Económica

El procedimiento en este caso es idéntico al caso anterior, salvo que el nodo *k* seleccionado, en vez de ser el más cercano a cualquiera de los puntos en el subgrafo, se escoge el nodo *k* tal que $c_{ik}+c_{kj}-c_{ij}$ es mínimo y entonces se inserta *k* entre *i* y *j*. Este procedimiento solamente es aplicable en las situaciones en las que $c_{ij} \neq d_{ij}$.

C: Inserción más Lejana

Es un método exactamente igual que el procedimiento de inserción más cercana salvo que en vez de escoger el c_{ik} mínimo se escoge el c_{ik} máximo para formar la ruta *i-k-i* y en segundo lugar, para considerar el punto nuevo, en vez de insertar el punto más cercano, se escoge el punto más lejano para su inserción.

D: Inserción más Rápida o Adición más Cercana.

En este caso el procedimiento es el siguiente:

1. Se escoge cualquier punto como comienzo del circuito *T*, con *1* nodo y *0* arcos.
2. Dado el circuito de *k* nodos T_k, buscar el nodo z_k que no esté en T_k que esté más cerca de un nodo. Denominarlo y_k en T_k.
3. Sea T_{k+1} el circuito de *k+1* nodos obtenido de la inserción de z_k inmediatamente después de y_k en T_k.
4. Repetir los pasos 2 y 3 hasta conseguir un ciclo Hamiltoniano.

E: Inserción de Parábola Convexa.

Construir la parábola convexa del conjunto de nodos. Esta parábola proporciona la primera subruta de nodos.

1. Para cada nodo *k* aún no incluido en la ruta, decidir entre que dos nodos de la misma se inserta, en función del mínimo coste de inserción $c_{ik}+c_{kj}-c_{ij}$
2. Para cada trío de nodos *(i, k, j)* encontrados en el paso anterior, determinar el (i^*, k^*, j^*) tal que $(c_{i^*k^*} + c_{k^*j^*}) / c_{i^*j^*}$ es mínima.
3. Insertar k^* entre i^* y j^*.
4. Repetir los pasos del 2 al 4 hasta construir un ciclo Hamiltoniano.

F: Inserción de Máximo Ángulo

1. Formar la parábola convexa en el conjunto de nodos.
2. Escoger el nodo k^* que aún no está en la subruta y el arco (i^*, j^*) en la ruta tal que el ángulo formado por los dos arcos (i^*, k^*) y (k^*, j^*) es el mayor posible.
3. Insertar el nodo k^* entre i^* y j^*.
4. Repetir los pasos del 2 al 3 hasta obtener un ciclo Hamiltoniano.

3.1.1.3 Enfoque de Mínimo Árbol Expandido

Consta de los siguientes pasos:

1. Buscar el mínimo árbol expandido *T* de *G*.

2. Doblar los bordes del árbol para obtener un ciclo Euleriano.
3. Retirar los polígonos de los nodos que tengan grado superior a *2* y transformar de esta manera el ciclo Euleriano en un ciclo Hamiltoniano.

3.1.1.4 Heurístico de Christofides

Este heurístico procede de la siguiente manera:

1. Buscar el mínimo árbol expandido *T* de *G*.
2. Identificar los nodos que tengan un grado impar en *T*. Resolver un emparejamiento de los nodos con grados impares con mínimo coste.
3. Añadir los nodos de la solución del emparejamiento a los nodos que ya están en *T*, obteniendo un ciclo Euler. En este subgrafo, cada nodo tiene grado par, aunque algunos tendrán grados superiores a *2*.
4. Retirar los polígonos de los nodos que tengan grado superior a *2* y transformar el ciclo Euleriano en ciclo Hamiltoniano.

3.1.1.5 Fusión más cercana

Este método persigue construir una secuencia de $S_1,\ldots,S_n$ donde cada S_i uno contiene un conjunto de $n\text{-}i+1$ subrutas disjuntas que cubren todos los nodos. Para ello se siguen los siguientes pasos:

1. S_1 consiste en n subrutas, cada una con un único nodo.
2. Para cada $i<n$, buscar un arco (a_i, b_i) tal que
3. $c_{a_i,b_i} = \min\{c_{xy} \ \ para\ x\ e\ y\ en\ diferentes\ subrutas\ de\ S_i\}$
4. Entonces S_{i+1} se obtiene a partir de S_i a través de la fusión de las subrutas que contienen a_i y b_i (en cada iteración se fusionan las dos subrutas que estén más cercanas).

Hasta este punto se han desarrollado los algoritmos de construcción de rutas, cuyas características generales se mencionaban al comienzo de la sección: rapidez, facilidad de aplicación y mediocridad de los resultados. Se trata de algoritmos primitivos y deterministas puesto que siempre alcanzan la misma solución, ya que sus reglas lo exigen así. La aplicación única de estos métodos proporciona rutas mediocres dado que distan en torno al 10% de los valores óptimos de las mismas. Por ello se han desarrollado una serie de métodos para mejorar estos resultados. Estos métodos de mejora de rutas se consideran a continuación.

2.4.2 Procedimientos de mejora de rutas

En este apartado se recogen los procedimientos que permiten la mejora de rutas a partir de una solución dada, ya sea proveniente de un método de construcción de rutas anteriormente aplicado, o bien una secuencia de nodos escogida aleatoriamente. Se trata de métodos que persiguen estudiar cómo varía la solución final si se producen una serie de intercambios en la secuencia de nodos. En concreto, los modelos más conocidos son

los de 2-opt y 3-opt de Lin (1966) y el *k*-opt para *k>3* de Lin y Kernighan (1973). Sin embargo son procedimientos que obtienen óptimos locales, de manera que el comportamiento de cada uno de los métodos es mejor a medida que se va incrementando el número de nodos intercambiados (*k*).

La forma de operar de este tipo de modelos consiste en que el algoritmo constantemente considere las operaciones de intercambio de dos (2-opt), tres (3-opt) o *k* (*k*-opt) arcos simultáneamente de manera que solamente se realizará el intercambio si este mejora la distancia total recorrida. Este proceso se repetirá hasta que la ruta ya no pueda ser mejorada, de manera que se habrá obtenido un óptimo local.

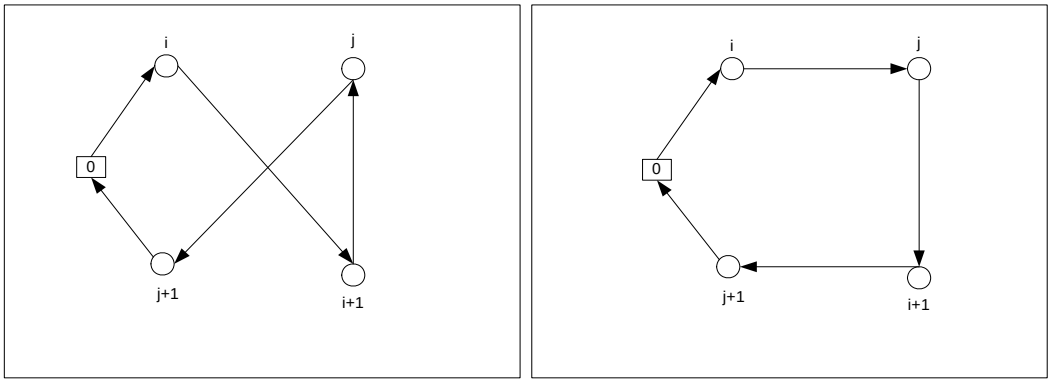


Figura 2.3 Ejemplo de un cambio 2-opt. Los arcos *(i, i+1)* y *(j, j+1)* se reemplazan por los arcos *(i, j)* y *(i+1, j+1)*

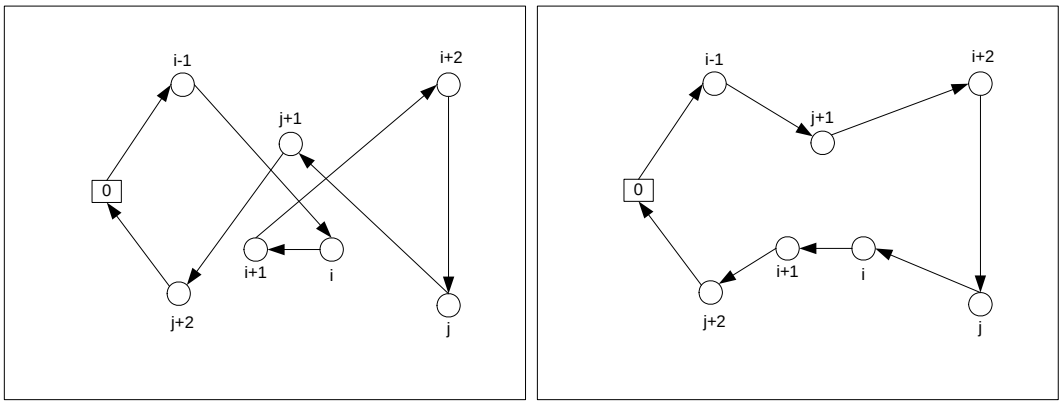


Figura 2.4 Ejemplo de un cambio 4-opt o de doble puente. Los arcos *(i-1, i), (i+1, i+2), (j, j+1)* y *(j+1, j+2)* se sustituyen por los arcos *(i-1, j+1), (j+1, j+2), (j, i)* y *(i+1, j+2)*

Este proceso de búsqueda también se conoce con el nombre de *neighborhood search process*, o proceso de búsqueda en el vecindario, consistente en que el algoritmo tiene en cuenta todas las soluciones derivadas de hacer todos los cambios posibles entre nodos tomados de dos en dos, de tres en tres o de cuatro en cuatro. De esta manera el vecindario de tipo 2, 3, ó 4, es un conjunto de soluciones en el que se contemplan todas las variaciones derivadas de hacer estas combinaciones.

Un mecanismo similar de inserción es el Or-opt presentado por Or (1976) en su Tesis Doctoral para el TSP. La idea principal es el reposicionamiento de una cadena de *l* vértices consecutivos (clientes), a través del intercambio de tres arcos en la ruta original por otros tres arcos sin modificar la orientación de la ruta.

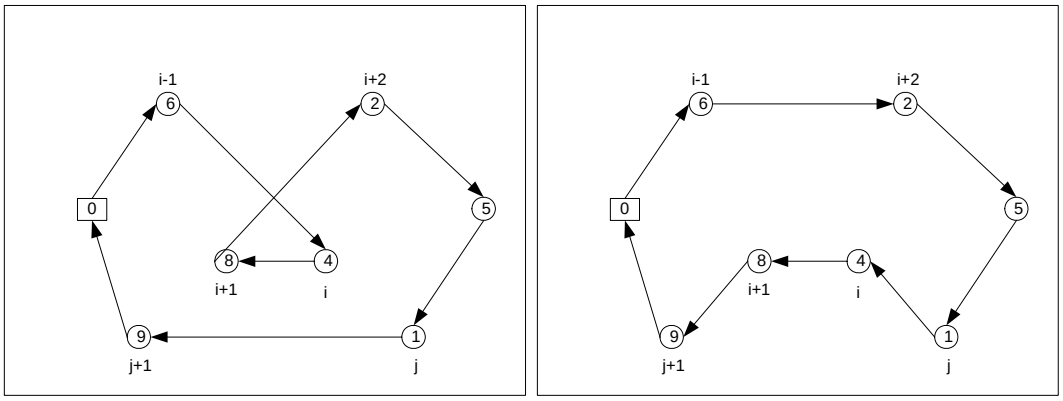


Figura 2.5 Funcionamiento del Operador de intercambio de Or. Los clientes *i* y *j* se reposicionan para ser visitados entre los clientes *j* y *j+1*. esto se consigue a través del intercambio de 3 arcos *(i-1, i), (i+1, i+2)* y *(j, j+1)* por los arcos *(i-1, i+2), (j, i)* y *(i+1, j+1),* manteniendo la orientación de la ruta

Uno de los inconvenientes de este tipo de métodos es el tiempo de cálculo utilizado, ya que cuanto mayor es el número de arcos considerados simultáneamente, mayor es el tiempo de computación. Además generalmente es necesario poner un límite artificial al número de iteraciones efectuadas, ya que si no el algoritmo no dejaría de procesar datos.

Una de las soluciones desarrolladas para disminuir el tiempo de cálculo para este tipo de algoritmos consiste en realizar preproceso de datos o incluso cálculos paralelos con varios procesadores al mismo tiempo. Por ejemplo, gran parte del tiempo de computación necesario para estos algoritmos se dedica a la construcción de las listas de ciudades a ser consideradas en los intercambios de nodos. Tradicionalmente se calculaba toda la lista de posibles candidatas. Sin embargo no es necesario que el procesador que elabora los intercambios, también tenga que elaborar la lista, por lo que estas dos tareas se pueden separar en procesadores diferentes, siempre que los procesadores sean capaces de almacenar toda la información del problema en sus memorias. Además cuando se empieza a calcular las posibles variantes, es posible que diferentes procesadores estén aplicando el mismo algoritmo sobre la misma lista de candidatos, de manera que los esfuerzos se multiplican y el tiempo de procesado se reduce drásticamente. Por ello, generalmente para grandes problemas (más de 1.000 nodos) es aconsejable la utilización de computación paralela en varios ordenadores, o bien en ordenadores multiprocesador.

Otros métodos para acelerar este tipo de algoritmos tienen en cuenta la partición del grafo (Rohe, 1995) para calcular las rutas en cada una de las particiones o subáreas consideradas de manera independiente y, posteriormente, integrar todas ellas en la conformación de la ruta.

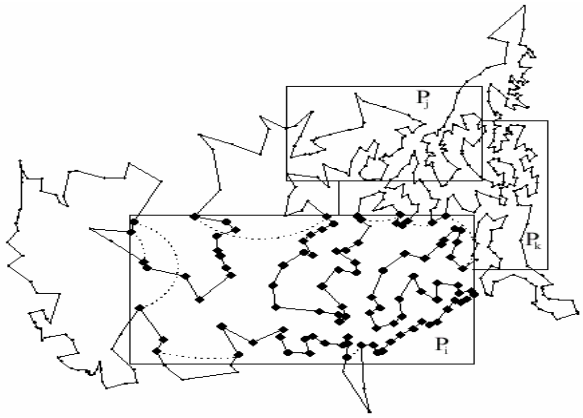


Figura 2.6 Enfoque de Rohe (1995) en la resolución del `att532`, se particiona al grafo y se calcula la ruta en esa partición para posteriormente unir las subrutas.

Otra forma para acelerar el proceso consistiría en considerar una única ruta generada de manera más o menos rápida y, sobre ella, utilizando múltiples procesadores para ir obteniendo mejoras a través de algunos de los métodos de *k*-opt, sin que en cada operación se pudiesen estorbar dos o más procesadores. Así se exploraría de manera mucho más rápida la solución considerada.

2.4.3 Procedimientos compuestos: Metaheurísticos

En el apartado anterior se prestaba especial atención a los métodos de aproximación basados en el intercambio de arcos para la mejora de las soluciones de partida. Generalmente todas las soluciones exploradas por estos métodos se integran dentro del denominado *k*-vecindario, de manera que el algoritmo selecciona la mejor solución de ese *k*-vecindario, donde *k* representa el número de arcos considerados de forma simultánea para el intercambio. Dado que todos estos algoritmos van seleccionando aquellas alternativas que mejoran las soluciones anteriores, se les ha denominado algoritmos de *Steepest Descent*, ya que de entre todas los intercambios que van verificando, siempre seleccionan aquellas soluciones que mejoran las de partida, de forma que habitualmente quedan estancados en óptimos locales.

Por esto las últimas evoluciones en los algoritmos heurísticos tratan de implementar estos métodos basados en intercambios, dentro de modelos matemáticos que fomenten la búsqueda en zonas diferentes, de manera que se eviten estos óptimos locales para explorar nuevas posibilidades, aunque estas empeoren las soluciones encontradas hasta ese momento.

En la figura 2.7 se recoge de manera más ilustrativa la situación de búsqueda del óptimo global. Es necesario destacar que existen multitud de posibles combinaciones (recogidas en el eje de abscisas) que a su vez proporcionan otras tantas soluciones (en el eje de ordenadas). De este modo, podemos representar la curva de costes, de manera que se puede observar como existen multitud de subóptimos, u óptimos locales. Tal y como se comentaba antes, el hecho de encontrar una buena solución y posteriormente intentar

mejorarla mediante intercambios de arcos puede resultar en una búsqueda dentro del vecindario del óptimo local y, aunque mejore la situación inicial, seguirá dentro del óptimo local. Por ello se han desarrollado el segundo tipo de algoritmos, conocidos como algoritmos metaheurísticos, entre los que destacan la Búsqueda Tabú (*Tabu Search*, TS), el Recocido Simulado (*Simulated Annealing*, SA) y los algoritmos genéticos entre otros.

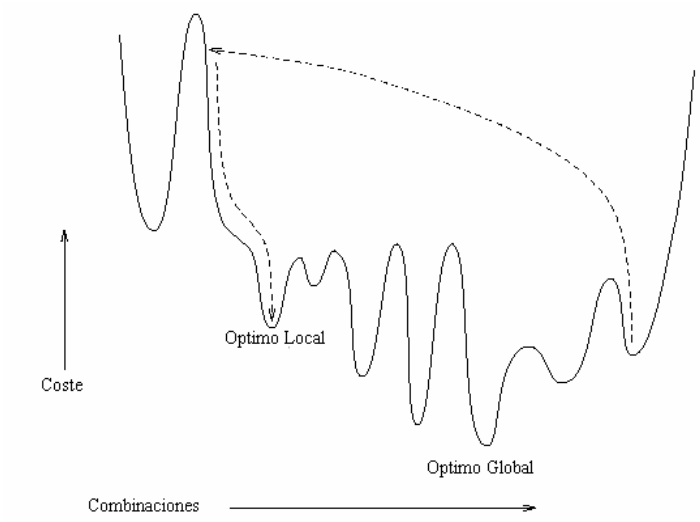


Figura 2.7 Mapa de los óptimos locales contra el óptimo global y las soluciones de los algoritmos de segunda y tercera generación.

Estos algoritmos incluyen generalmente rutinas o instrucciones que obligan a abandonar una determinada solución para probar con otras soluciones nuevas. Las diferencias entre todos ellos se exponen a continuación.

3.1.1.6 Búsqueda Tabú

La búsqueda tabú proviene de la idea de que no todas las soluciones de óptimos locales resultantes de técnicas como las de 2-opt ó 3-opt son buenas. Por ello es interesante modificar estos algoritmos de búsqueda local a través de alguna instrucción que provoque que la búsqueda se escape de dichos óptimos y lo incite a ir a otras áreas todavía sin explorar, tal y como se expone en la figura 2.7.

Una posible solución para ampliar la búsqueda consistiría en utilizar un algoritmo que comenzase desde diferentes puntos el proceso de optimización de manera que se contemplarían multitud de posibilidades. Para ello sería necesario repetir múltiples veces la búsqueda y contrastar las diferentes soluciones. Este enfoque de comienzo aleatorio y repetitivo puede perder eficacia si tenemos en cuenta que los óptimos suelen coincidir con soluciones más o menos próximas, de manera que si repetimos las búsquedas de manera aleatoria, estaríamos perdiendo los buenos resultados de experiencias previas. Por ello la búsqueda tabú persigue aprovechar la información de las soluciones vecinas para obtener una mejor aproximación. Para ello contempla posibles avances en la solución admitiendo

movimientos que se alejen de la mejor solución encontrada hasta el momento. Para ello actúa de la siguiente manera.

Suponiendo que todas las soluciones vecinas se han probado, la búsqueda tabú propone partir este óptimo local e identificar la mejor solución vecina (un nuevo óptimo local), que se utilizará como punto de partida para la posterior iteración, incluso aunque ésta empeore la situación inicial. Si esto fuese así, estaríamos ante un planteamiento similar al de los algoritmos de *k*-opt. Sin embargo la búsqueda tabú tiene en cuenta los últimos movimientos considerados, para no recaer en el óptimo inicial, de manera que obliga al algoritmo a explorar soluciones todavía no consideradas, es por ello que este tipo de movimientos se declaran movimientos prohibidos, o simplemente movimientos *"tabú"*.

Generalmente trabajan todos sobre movimientos 2-opt y se incluyen en la lista prohibida (tabú) aquellos movimientos o arcos ya considerados por el algoritmo. Esta lista tabú consiste en pares de arcos, donde cada par se corresponde con los arcos considerados en movimientos anteriores del 2-opt. En este sentido, un movimiento es tabú o prohibido si reinserta un par de arcos que están en la lista.

Por otro lado se consideran los *niveles de aspiración*. Los niveles de aspiración son los que hacen referencia a cuán permisivo es el algoritmo a la hora de reinsertar unos arcos que ya están en la lista tabú. Estos permisos se conceden generalmente para reintegrar pares de arcos que están en la lista tabú siempre que su reintegración suponga una reducción de la distancia de la ruta en menor medida de la distancia que suponía cuando originalmente estos arcos estaban en la ruta.

Existen variantes en los planteamientos como por ejemplo el algoritmo de Troyon (1988) y Malek, o Guruswamy y Pandya (1989) que se basan en las ciudades a las que hacen referencia los arcos, en vez de los arcos per se, aunque la operativa del algoritmo es la misma.

Voudouris y Tsang (1995) han ensayado con los problemas del TSPLIB para los casos con reducido número de nodos, obteniendo en un 99% de los casos la solución del límite inferior propuesta por Held y Karp (1970 y 1971), sin embargo se trata de los ejemplos con un número máximo de 150 nodos, por lo que lograr soluciones próximas al óptimo no resulta difícil.

Generalmente los algoritmos de búsqueda tabú requieren un tiempo de computación de N^3, N^4 o incluso algunos de hasta N^6, por lo que resultan ineficientes para problemas que contengan un elevado número de nodos y por ello no pueden competir con los algoritmos de tipo 2-opt y 3-opt. Por ejemplo, Glover (1989); Malek, Guruswamy y Pandya (1989) y Knox (1994) restringen la aplicación de sus algoritmos a problemas de menos de 110 nodos, dado el tiempo de cálculo necesario. Consecuentemente han quedado relegados a un segundo plano y aunque obtienen resultados sensiblemente mejores que los algoritmos de intercambio sencillos, su mejoría no justifica el tiempo de computación necesario.

A pesar de ello y al igual que ocurría en los casos de intercambio de arcos, es posible aplicar computación paralela para disminuir los tiempos de cálculo y así hacer más operativa la búsqueda tabú.

3.1.1.7 El algoritmo de Lin-Kernighan (1973)

Este fue el algoritmo considerado de mejor calidad en cuanto a resultados y tiempo de computación durante aproximadamente una década y media, desde 1973 hasta 1989, que es cuando se plantea la aplicación de los algoritmos genéticos para la resolución de los problemas TSP.

Se trata de un algoritmo de búsqueda local, que emplea movimientos de tipo 2-opt, pero considerando un conjunto de nodos mucho más restringido que en el típico 2-opt y que en el caso de la búsqueda tabú. Por ello existen múltiples criterios de restricción de la lista de nodos candidatos para ser considerados como movimiento de 2-opt.

Para ello los autores proponen que la solución obtenida inicialmente ha de considerarse como un camino Hamiltoniano y no como un circuito cerrado. Además, este camino está anclado por uno de los nodos extremos, que denominamos t_1. Además también consideramos el otro extremo del camino en la iteración *i*, que denominamos t_{2i}. En este sentido se ha configurado el camino Hamiltoniano entre todos los nodos, de manera que la ruta se conseguiría uniendo los dos extremos.

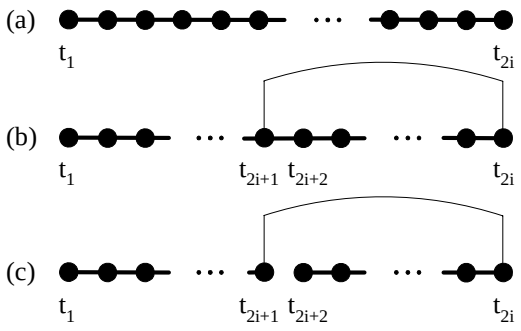


Figura 2.8 Algoritmo de Lin-Kernighan (1973)

El camino con esos dos extremos puede ser modificado a través del proceso de búsqueda de LK. En cada iteración solamente se consideran movimientos 2-opt, de manera que siempre uno de los arcos a considerar como intercambiable es el arco que uniría los dos extremos del camino. Además este cambio realizado ha de proporcionar una mejora en el camino Hamiltoniano de manera que se reduzca la distancia total. Para ello generalmente se construye la lista de posibles candidatos para realizar este movimiento de 2-opt y que dadas sus características tan restrictivas, supone la exclusión de muchas posibilidades. En este sentido la lista de candidatos se reduce a todos aquellos movimientos que proporcionen una distancia menor a *k*, de manera que solamente se considerarían los

nodos vecinos a la alternativa en consideración. De esta forma se van efectuando las uniones que mejoran las soluciones de partida.

Para realizar estas uniones se consideran dos listas: una con los movimientos prohibidos (como en el caso de la búsqueda tabú) y otra con los arcos añadidos. Un movimiento es tabú si intenta o bien incluir un arco de la lista de los arcos eliminados o bien borrar un arco de la lista de los arcos añadidos.

Al contrario que lo que ocurre en la búsqueda tabú, en el algoritmo de LK no es posible conceder permisos para reintegrar arcos ya eliminados. En este sentido el algoritmo de LK finaliza cuando ya no existen movimientos que sean posibles para la ruta en consideración.

Los movimientos que se consideran posibles y que no rompen ninguna de las restricciones del algoritmo, se prueban para establecer cuáles de ellos mejoran las soluciones de partida. Estos movimientos han de proporcionar por lo tanto una ruta de menor longitud que la ruta en consideración. Por ello se guardan para posterior exploración. Sin embargo el movimiento que finalmente se escoge es aquel que proporciona un camino menor, lo cuál no implica que este camino nuevo sea mejor que el camino que existía en la solución de partida, ni que la ruta resultante después del cambio sea mejor que la anteriormente considerada.

La ventaja de este algoritmo es que reduce mucho el tiempo de computación al establecer tantas restricciones sobre las posibilidades de arcos intercambiados de cada vez. Esta reducción de tiempo generalmente implica una penalización en las soluciones obtenidas ya que se limita mucho las posibilidades de optimización local. Sin embargo esta limitación en las posibilidades en cada movimiento LK se ven compensadas por la posibilidad de aplicar multitud de comienzos diferentes sobre las rutas iniciales, lo cual implica gran intensificación en el cálculo.

El algoritmo utiliza una serie de fases basadas en esta intensificación sobre la *ruta ganadora*, consistente en le mejor ruta hasta el momento. Inicialmente esta ruta campeona proviene de cualquier heurístico de construcción de rutas. Cuanto mejor sea el heurístico, mejor solución proporcionará el LK. Sobre esta solución inicial se comienzan a considerar todos los movimientos de tipo 3-opt y 4-opt, intensificando la búsqueda sobre los alrededores de la ruta campeona hasta el momento. Cuando se halla una ruta que es mejor, se completa la búsqueda y se considera como nueva campeona. Una vez que se obtiene una nueva ruta campeona, entonces se vuelven a aplicar los algoritmos de mejora 3-opt y 4-opt. De esta manera y considerando todas las alternativas posibles para un determinado camino Hamiltoniano, se escoge aquella alternativa que ha proporcionado la mejor solución, de manera que el algoritmo de mejora se detiene cuando no existen más alternativas, es decir cuando halla una solución que supone un óptimo local con respecto al vecindario adyacente.

Recientemente se han intensificado los trabajos basados en la búsqueda tabú y en el algoritmo de Lin-Kernighan (1973). Existen multitud de variantes, sobre todo a la hora de establecer las restricciones del algoritmo de LK. Cada autor propone sus propios criterios para considerar los nodos a intercambiar, pero generalmente el planteamiento es el mismo que el desarrollado por LK.

Zachariasen y Dam (1995), proponen como estructura alternativa al camino Hamiltoniano, una estructura similar a la de una flor, incluyendo la *raíz*, el *tallo* y un ciclo al final del mismo, a modo de *capullo*. De esta manera los cambios posibles han de obedecer a esta estructura. El planteamiento es muy similar al de LK, pero con la variante de la estructura determinada que proponen estos autores.

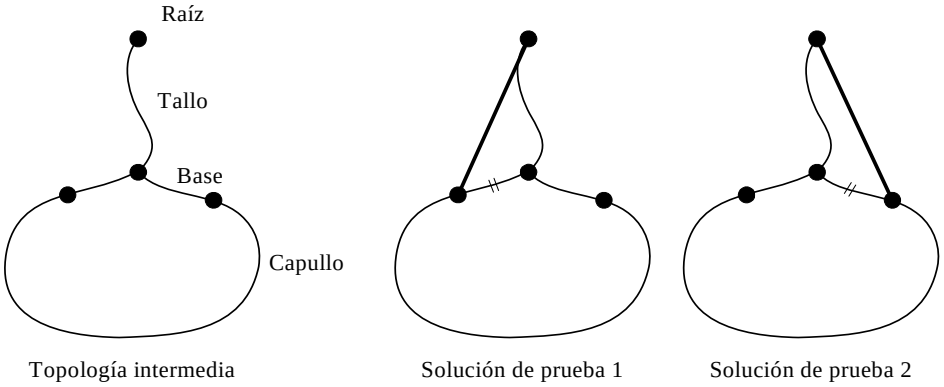


Figura 2.9 El algoritmo de la Flor

Martin et al. (1991 y 1992) proponen una combinación del algoritmo de LK con heurísticos de búsqueda local para realizar una segunda fase de optimización generalmente a través de un refinado de la solución final con posteriores iteraciones del método 2-opt, algo que en la literatura se conoce como el algoritmo Iterativo de LK y que posteriormente revisamos. La aplicación de este método combinado mejora las soluciones obtenidas por el algoritmo básico de LK.

Actualmente se siguen desarrollando investigaciones sobre este tipo de algoritmos ya que se consiguen soluciones muy próximas al óptimo. Para problemas con un gran número de nodos (más de un millón) se están obteniendo soluciones que sobrepasan solamente entre un 1 y 2% el límite inferior propuesto por Held y Karp (1970 y 1971).

3.1.1.8 Recocido simulado (Simulated Annealing o SA)

Se trata de otra de las técnicas de optimización local basada en los movimientos de tipo 2-opt para acercarse al óptimo local (*downhill moves*). En el caso del SA se permite la realización de movimientos de alejamiento (*uphill moves*) en cualquier momento, de manera completamente aleatoria, no como en el caso de TS, donde solamente se permitían los movimientos cuando el algoritmo se quedaba atascado en la búsqueda de óptimos locales. La diferencia con el TS es por lo tanto su grado de aleatoriedad en la

búsqueda de mejores soluciones, ya que no sigue ninguna pauta preestablecida, como ocurre en el caso de TS.

El algoritmo original fue desarrollado por Kirkpatrick, Vecchi y Gelatt (1983) y Cerny (1985), aunque la búsqueda aleatoria se debe a Metropolis et al. (1953) que trata de simular el movimiento de los átomos en un baño de galvanizado. Incluyen un parámetro de temperatura de manera que va disminuyendo a medida que el algoritmo realiza su búsqueda, simulando el proceso de galvanizado.

El esquema general es el mismo para casi todos los trabajos de esta línea de investigación. Sin embargo los diferentes autores presentan multitud de técnicas para generar la ruta inicial considerada por el SA. A medida que la temperatura va descendiendo el alcance de los movimientos considerados por el algoritmo va descendiendo, centrándose cada vez más en la búsqueda local en perjuicio de la búsqueda más lejana. El esquema general de este tipo de algoritmo es el siguiente:

1. Generar una solución inicial S y considerarla como la solución campeona $S^* = S$.
2. Determinar la temperatura inicial T.
3. Mientras no haya congelación:
   - 3.1. Mientras no haya equilibrio para esta temperatura, realizar lo siguiente:
     - 3.1.1. Escoger una solución vecina aleatoria S'
     - 3.1.2. Marcar $\Delta = Longitud(S') - Longitud(S)$.
     - 3.1.3. Si $\Delta \leq 0$ (movimiento downhill):
       Fijar $S = S'$
       Si $Longitud(S) < Longitud(S^*)$, fijar $S^* = S$.
     - 3.1.4 Caso contrario (movimiento uphill):
       Escoger un número aleatorio r de manera uniforme en $[0\ ,\ 1]$.
       Si $r < e^{-\Delta/T}$, fijar $S = S'$
     - 3.1.5 Fin del bucle ‘‘Mientras no haya equilibrio”.
   - 3.2 Bajar la temperatura T.
   - 3.3 Fin del bucle ‘‘Mientras no haya congelación”.
4. Devolver S^*.

Mediante el modelo de las cadenas de Markov se ha probado que la solución del SA es convergente al óptimo. Sin embargo el tiempo necesario sería similar al empleado para hacer una búsqueda exhaustiva. En la práctica, la interpretación es diferente y las reducciones de temperatura no obedecen al modelo de las cadenas de Markov, que haría altamente ineficiente su aplicación. Por el contrario, las aplicaciones prácticas de este algoritmo generalmente consideran bajadas de temperatura próximas a la unidad y además un determinado número de movimientos para cada temperatura cercanos a 50.

Para adaptar el SA al TSP, es necesario realizar movimientos de tipo *k*-opt, en concreto el 2-opt (Kirkpatrick et al. (1983) y Cerny (1985)). Además es necesario establecer el número de movimientos a considerar para cada temperatura, pues para obtener buenas soluciones este número ha de ser proporcional al tamaño del vecindario. También se

necesita definir el rango de temperaturas empleado y el valor de cada bajada de temperatura.

Uno de los inconvenientes del SA es que para obtener buenos resultados es necesario que el rango de temperaturas sea lo mayor posible y que el número de iteraciones también sea grande. Recordemos que se trata de un algoritmo asintóticamente óptimo. Sin embargo al incrementar el número de cálculos también se incrementa de forma exponencial el tiempo de computación sin que los resultados obtenidos sean excepcionalmente buenos comparado con algoritmos mucho más sencillos como el 3-opt, o el LK. Por ello es necesario establecer instrucciones o rutinas que permitan acelerar el proceso.

Bonomi y Lutton (1984) proponen dos ideas clave para acelerar el algoritmo. En primer lugar lo que se denomina el recorte del vecindario (*Neighborhood Pruning*). Consiste en excluir de los movimientos de tipo 2-opt aquellos que impliquen introducir un arco en la solución que sea demasiado grande, pues esto sólo conseguirá empeorar la solución. Por ello buscamos un mecanismo que elimine estos posibles movimientos a priori. El mecanismo desarrollado por estos dos autores consiste, en dividir el cuadrado de menor tamaño que incluya a todas las ciudades, en una cuadrícula de dimensiones más pequeñas. Suponiendo que las ciudades estén distribuidas uniformemente, a la hora de considerar las alternativas en un movimiento 2-opt, solamente se consideraran aquellas que impliquen a ciudades en la misma cuadrícula que la ciudad inicial, de manera que el número de computaciones se reduce considerablemente.

Además los autores proponen una segunda idea para limitar la búsqueda, consistente en comenzar con bajas temperaturas, dado que si se consideran bajas temperaturas, los posibles movimientos *uphill*, o de largo alcance, van a estar más limitados. Además sugieren el comenzar con una solución predeterminada, generalmente a través de un heurístico que sea muy rápido como por ejemplo el *Nearest Neighbor*.

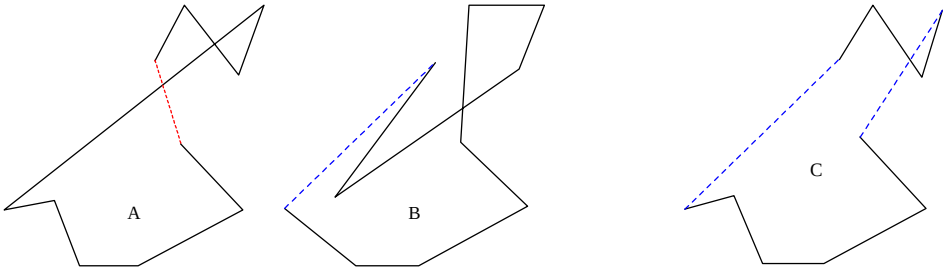
Aún así los resultados obtenidos por Kirckpatrick et al. (1983), con la aplicación de estas técnicas dejan mucho que desear si los comparamos con los de LK, o los sencillos 2-opt y 3-opt. Los resultados mediocres conseguidos, incluso con la aplicación simultánea de ambas ideas por Bonomi y Lutton (1984), se deben a una de las hipótesis de partida del algoritmo –la distribución uniforme de las ciudades o nodos– algo que no ocurre con asiduidad en los problemas tipo de la literatura y por eso generalmente se obtienen peores resultados que otros algoritmos que no se basan en dicha uniformidad.

A pesar de ello, en los últimos años y utilizando la combinación del esquema de SA, con técnicas de búsqueda local mucho más potentes, como son el 3-opt, o el propio LK, se han obtenido resultados muy buenos, cercanos a los conseguidos con los algoritmos de búsqueda tabú o con los algoritmos genéticos, que revisamos en la siguiente sección. Es por ello necesario combinar estas herramientas con técnicas que mejoren los movimientos de búsqueda local.

Un ejemplo de este tipo de trabajos es el realizado por Wendt (1995) y denominado COSA (*Co-Operative Simulated Annealing*). En este planteamiento se combinan tanto los principios del SA como la potencia de los operadores de intercambio provenientes de los algoritmos genéticos.

COSA aplica una implementación sincronizada de múltiples procesos de SA, conjuntamente con transiciones cooperativas. Estas transiciones cooperativas reemplazan la función de probabilidad uniforme que utiliza el SA para generar una solución vecina *j* de una solución *i*. En vez de esta función de probabilidades uniforme, se utiliza una función probabilística de una sola cola. Es decir, que la utilización de esta función implica que ciertos nodos tienen mayor probabilidad de ser escogidos para formar parte de una determinada solución. En cierta manera es equivalente a que estos nodos ejerciesen una fuerza de atracción de manera que provoca que la solución se incline por incluir o acortar la distancia hacia estos nodos, en vez de considerar aquellos que alejarían la solución incrementando la distancia.

Mientras que en el clásico SA los arcos para realizar el 2-opt se obtienen de manera aleatoria, en este método se aprovecha la información relevante para la realización de la elección de los arcos intercambiados. En este sentido la elección del primer arco es completamente aleatoria. En la figura 2.10 se ha escogido el arco que aparece en rojo, de la solución *A*. A continuación, en vez de seleccionar el segundo arco también de forma aleatoria, lo que se realiza es la elección del arco que sustituirá al primero y que en cierta forma condiciona la elección del segundo a eliminar, dado que ha de contar con uno de los extremos incluidos en el primer arco.



Dos soluciones individuales del TSP

Solución C obtenida a través de un movimiento de transición y de cooperación

Figura 2.10 Resolución del TSP a través del algoritmo COSA

Para seleccionar el arco que sustituirá al arco eliminado se tiene en cuenta la cooperación. Esta cooperación se basa en la afirmación de que existe información suficiente en otras soluciones como para establecer el arco que ha de ser seleccionado. Por ello se selecciona de forma aleatoria otra solución *B* y comprobamos en la misma qué arco deja una de las dos ciudades de *B* en condiciones de necesitar un nuevo arco en la solución *B*. Supongamos que *B* es la solución escogida que aparece en el gráfico y que la línea punteada es una posible respuesta. Habiendo escogido un arco de *A* de forma aleatoria para ser eliminado y un nuevo arco para ser insertado en *A* a través de la cooperación,

entonces no queda lugar a dudas de cuál es el segundo arco que ha de ser eliminado y su consecuente sustituto. De manera que la solución sería la *C*.

A continuación y al igual que ocurre en el SA, la aceptación o rechazo del movimiento depende de la función de probabilidad incluida en el algoritmo general. En caso de aceptar, la solución nueva *C* está un poco más cercana a *B*, dado que ahora comparte más arcos comunes que al principio. Es por ello que al igual que ocurre en los algoritmos genéticos las soluciones individuales convergen.

Una de las ventajas del SA es que se manipula fácilmente dado que se trata de un algoritmo muy parametrizado y por ello muy flexible. Permite jugar con diferentes variables para adaptar el algoritmo a los diferentes casos, de ahí que los resultados obtenidos y el tiempo de computación dependan en gran medida de cómo se fijen los parámetros. La aplicación del SA ha evolucionado mucho hasta alcanzar niveles cercanos a los mejores algoritmos genéticos. Además en la actualidad se trabaja en la combinación y recombinación de diferentes algoritmos para conseguir mejoras globales.

3.1.1.9 Algoritmos genéticos

El planteamiento de los algoritmos genéticos para la optimización tiene lugar en los años 70, aunque su aplicación a la resolución de TSPs no tiene lugar hasta la década de los 80. La ventaja de este tipo de algoritmos es que presentan una forma de generar soluciones nuevas y mejoradas (*hijos o descendientes*) partiendo de soluciones previas (*padres*). Es necesario tener en cuenta que las operaciones para construir soluciones de partida pueden realizarse basándose en procesos paralelos, que posteriormente se funden en nuevas soluciones de partida. Esto es lo que se conoce como Algoritmos Genéticos Paralelos. Para una revisión en mayor profundidad de los fundamentos, hipótesis y aplicaciones de los algoritmos genéticos es recomendable revisar el trabajo de Laguna y Moscato (1995). Merece la pena comentar que las diferentes técnicas de los algoritmos genéticos provienen de la teoría de Darwin sobre la evolución de las especies. Estas técnicas evolutivas se centran en cuatro principios básicos:

- Selección
- Mutación
- Recombinación
- Aislamiento

Las aplicaciones más inmediatas derivan de los tres primeros factores y casi en su mayor parte de los que implican selección y combinación. La mutación es un factor que tardó en ser aplicado algunos años como veremos a continuación y finalmente el aislamiento se considera a mediados de los 90, siendo introducido por Gorges-Schleuter (1991).

A la hora de aplicar los algoritmos genéticos al caso de los TSP es necesario definir una serie de consideraciones previas.

- El valor *k* de los intercambios de nodos

- Los métodos para generar soluciones iniciales.
- El algoritmo de optimización local
- La estrategia de emparejamiento
- La naturaleza de los cruces entre soluciones
- La estrategia de selección
- El criterio de convergencia.

El algoritmo genético genérico funciona de la siguiente manera:

1. Generar una población de k soluciones iniciales $S = (S_1, \ldots, S_k)$
2. Aplicar un algoritmo de optimización local a cada solución S_i de S de manera que se reemplace la mejor solución de S
3. Mientras no haya convergencia, hacer lo siguiente:
   - 3.1. Seleccionar k' subconjuntos disjuntos de S de tamaño *1* ó *2* como padres (estrategia de apareamiento).
   - 3.2. Para cada subconjunto de *1* elemento, realizar una mutación aleatoria para obtener una nueva solución.
   - 3.3. Para cada subconjunto de *2* elementos, realizar una operación de cruce (posiblemente aleatoria) para obtener una nueva solución que contenga pautas de ambos padres.
   - 3.4. Aplicar un algoritmo de optimización local a cada una de las k' soluciones producidas en 3.3. Sea S' el conjunto de las nuevas soluciones.
   - 3.5. Utilizando una estrategia de selección, escoger los k supervivientes de $S \cup S'$ y reemplazar las soluciones iniciales de S con estos supervivientes.
4. Proporcionar la mejor solución de S

A continuación se recogen de manera más detallada algunos algoritmos genéticos desarrollados en la literatura para la resolución de los TSPs.

2.4.3.1.1 El algoritmo de Brady (1985) y sus sucesores

Brady utilizó el algoritmo de 2-opt para realizar el conjunto de rutas que posteriormente trataría de casar para ir obteniendo soluciones mejores. Sobre estas rutas, el autor se ciñó a la forma de emparejar estos resultados tratando de buscar padres en los resultados. Buscaba pautas de comportamiento similares en las diferentes rutas. De esta manera encontraba pequeños caminos que se repetían en las soluciones, con conjuntos de las mismas ciudades pero en diferente orden. Estos subconjuntos eran seleccionados y utilizados para su postoptimización. Así se conseguían mejores resultados en las rutas finales que realizando múltiples comienzos del 2-opt desde diferentes puntos aleatorios.

Trabajando con buenas soluciones previas, los resultados se conseguían en un breve espacio de tiempo comparado con otro tipo de algoritmos. Este algoritmo tan sencillo e intuitivo tuvo gran éxito dado su enorme potencial de mejora. Para problemas de unos 64 nodos era capaz de encontrar caminos comunes de hasta 32 nodos pero estos disminuían si se incrementaba el tamaño del problema.

Para problemas de entre 100 y 200 nodos, Suh y Van Gucht (1987) proponen un algoritmo de emparejamiento de soluciones consistente en una combinación del algoritmo del nodo más cercano con el algoritmo genético. Para conjuntos de 100 soluciones iniciales, conseguían resultados similares a los del heurístico 3-opt aunque solamente se mostraba eficiente para problemas de hasta 200 nodos, ya que sus tiempos de computación eran muy elevados comparados con los tiempos necesarios para el 3-opt.

3.1.1.9.1 El algoritmo de Mühlenbein et al. (1988)

En 1988 Mühlenbein et al. introducen un cambio en los algoritmos genéticos. Estos investigadores desarrollan un algoritmo para ser implementado de forma paralela. Se plantea que cada ruta se asigna a un procesador diferente, de manera que cada procesador comprueba su ruta con las rutas de los otros procesadores. Las rutas no se estudiaban por completo sino que se escogían de forma aleatoria algunas subrutas de tamaños que oscilaban entre 10 nodos hasta *N/2* nodos. Éstas se iban agregando hasta formar una nueva ruta.

Igualmente se incluye un mecanismo para acelerar la optimización local posterior bloqueando las ciudades clave que incluídas en los padres de cada ruta. De esta forma, cuando se aplican los procedimientos de postoptimización, se prohíbe que el algoritmo rompa estas uniones predeterminadas.

La postoptimización que realizaban se basaba en Or-opt y en 2-opt, aunque finalmente decidieron aplicar únicamente la 2-opt ya que les permitía ahorrar mucho tiempo de computación sin perjudicar los resultados, ya que podían dedicar más tiempo a la fase de generación de rutas.

Las máquinas utilizadas por estos autores se basan en un sistema de 16 procesadores *Encore* y el problema más grande que consiguen resolver es el `pcb442` del TSPLIB consiguiendo una solución de un 1% más corta que la media de los 3-opt y sólo un 0.1% más grande que la solución del LK. Posteriormente consiguieron mejorar estos resultados utilizando una lista truncada de vecinos para el 2-opt y un sistema de 64 *Transputers* con una solución para el `att532` del 0.19% superior al óptimo y una media del 0.94 % de mejoría con respecto al LK. El tiempo de computación en la red de 64 procesadores era de 3 horas, mientras que el LK en el *SGI Challenge* era de 2 segundos.
La discusión se centra ahora en si este algoritmo es mejor o peor que el LK. Desde luego que supone un gran competidor, a pesar de que el tiempo utilizado es mucho mayor que en el algoritmo LK. En este sentido para el mismo tiempo es posible aplicar múltiples veces el LK sobre el problema y conseguir un mejor resultado que el algoritmo genético.

3.1.1.9.2 El algoritmo de Martin, Otto y Felten (1991 y 1992) y el algoritmo Iterativo de Lin-Kernighan

Como conclusiones de lo anteriormente expuesto, para un tiempo determinado de computación, es posible que un algoritmo genético mejore sus resultados si se mejora el sistema de optimización local del mismo, al igual que si se incrementa el tamaño de la

población de soluciones. A primera vista parece que la característica más importante es la necesidad de mejorar el algoritmo de optimización local y no la de incrementar el tamaño de la población inicial de soluciones. Así, por ejemplo, Martin, Otto y Felten (1991, 1992) diseñaron un algoritmo genético que partía de una población de soluciones de tamaño 1, es decir, de una única solución.

En este algoritmo no es posible emparejar ninguna solución, de manera que sólo son posibles las mutaciones para obtener soluciones diferentes. Estas mutaciones se basan en cambios locales en la cadena de nodos que se producen como alteraciones aleatorias, permaneciendo constante el resto de la cadena. Las mutaciones en este algoritmo consisten en movimientos 4-opt de doble puente, algo no considerado ni por el 3-opt ni por el algoritmo de LK. Estos movimientos se escogen de un conjunto más restringido de posibilidades en los que ningún arco añadido puede exceder una longitud predeterminada, generalmente un pequeño múltiplo de la media de las distancias entre los nodos más cercanos.

La sustitución depende de si la solución obtenida es más corta o no que la original, y en el caso de que no lo sea, se aceptará con una probabilidad de $e^{-\Delta/T}$, donde Δ es el incremento de distancia con respecto a la solución de partida, igual que en el caso de SA, y T es la temperatura inicial seleccionada. Debido a este proceso aleatorio, los autores denominan su método como *Large Step Markov Chain Algorithm.*

Para la post-optimización local, Martin et al. (1991, 1992) también consideran congelar determinadas uniones para que el algoritmo, en este caso el 3-opt, no las desuna. Sin embargo en ciertos casos es permisivo para estas desuniones algo que no ocurría en el algoritmo de Mühlbein et al. (1988).

Los resultados para este algoritmo fueron prometedores. Por ejemplo, para el caso de `att532,` conseguían resultados un 0.07% superiores al óptimo en 15 horas, aunque posteriormente se mejoró a través de la utilización de una versión del algoritmo de LK para la fase de postoptimización en vez del 4-opt. Con resultados iguales al óptimo en unas 3 horas.

Con esta misma idea, Johnson (1990) utiliza el LK en su algoritmo genético, en una versión simplificada del algoritmo de Martin et al. y lo denomina el algoritmo Iterativo de Lin-Kerninham (ILK en adelante), donde cada iteración se corresponde a una generación de ruta utilizando la nomenclatura de los GAs.

Las mejoras de Johnson (1990) consisten en la eliminación de la restricción de distancia en el cambio 4-opt, de manera que amplia el abanico de posibilidades para la mutación. Además elimina la aleatoriedad del criterio de aceptación y sólo escoge aquellos cambios que acortan la distancia. Johnson (1990) resuelve por primera vez los problemas `lin318`, `pcb442`, `att532`, `gr666`, `pr1002` y `pr2392`, aunque no siempre obtienen las soluciones óptimas.

Actualmente el enfoque de Martin et al. y en concreto el ILK son los métodos más eficientes en cuanto a consumo de tiempo y resultados y sirven como referencia para los trabajos de Applegate, Bixby, Ch'vatal y Cook (1994) quienes han obtenido el record de los últimos TSPs de la TSPLIB resueltos hasta el óptimo (3.038, 4.461 y 7.397 nodos). Johnson, McGeoch y Rothberg (1996) han utilizado el ILK para confirmar que por término medio el límite inferior de Held y Karp permanece a una distancia de 0.75% del óptimo verdadero para problemas de 100.000 nodos por término medio. Estas implementaciones implican meses de computación en redes de procesadores.

Freisleben y Merz (1996) proponen un algoritmo basado en la creación de rutas a través del *Nearest Neighbor*, una posterior mejora a través de LK y a continuación la aplicación del GA mediante la realización del cruce buscando pautas de comportamiento comunes y considerando a los padres, de manera que la distancia entre el hijo y cada uno de los padres sea la misma, entendiendo la distancia como el número de nodos que están contenidos en una ruta y que no están en la otra. Esto lo denominan “cruce preservando la distancia“ (*distance preserving crossover, DPX*) y obedece a los estudios de Boese (1996), quien establece que la distancia entre los óptimos locales y el optimo global es por término medio la misma. En este sentido sabiendo que el LK halla óptimos locales, se deja libertad al algoritmo para que pueda buscar el óptimo global dejando esa distancia entre el hijo y sus padres como si se tratase de unos grados de libertad. Finalmente aplica un criterio de postoptimización a los resultados consistente en la mutación basándose en el 4-opt o de doble puente sin restricción de distancia. Actualmente está considerado como el mejor algoritmo genético con optimización local.

Hoy en día los avances en el campo de los GAs se centran en la computación paralela de las soluciones y la obtención de los hijos. El procesado en paralelo puede desarrollarse de múltiples formas, bien de manera independiente en diferentes procesadores o bien de manera inteligente en cada procesador, de manera que los algoritmos van mejorando de forma individual cada una de las soluciones propuestas en los procesadores independientes. Así, Moscato y Norman (1992) describen la aplicación de un PGA (*Parallel Genetic Algorithm*) en una red de procesadores mediante hardware de comunicación entre procesadores o de mensajes entre los mismos (implementación con transputers).

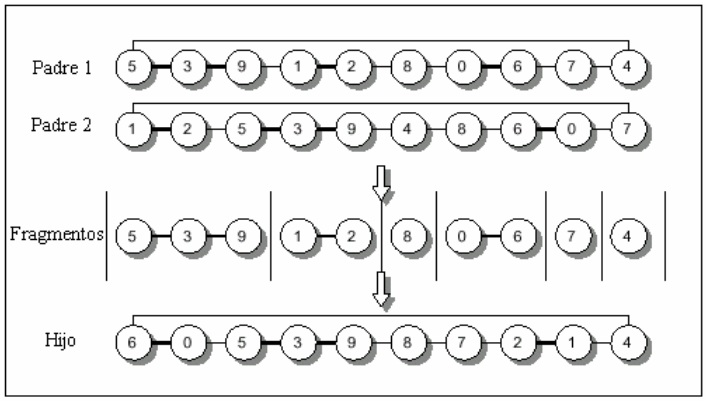


Figura 2.11 Cruce preservando la distancia

Las últimas y más ambiciosas aplicaciones hablan de problemas de 10 millones de nodos y hasta de 18 millones de nodos, correspondiéndose con la localización de otras tantas estrellas en el firmamento. Los objetivos de estos ensayos no son otros que buscar mejoras en la utilización de la información para conseguir reducciones en tiempos de computación. Así Rohe (1995), aplicó 4 estaciones de trabajo IBM durante dos semanas para resolver el problema de los 10.907.064 nodos, consiguiendo unos resultados de un 1.57% sobre el límite inferior de Held y Karp (1970 y 1971). De igual manera Applegate y Cook (1994) utilizaron una red de 10 estaciones de trabajo para resolver dicho TSP, recurriendo a su partición en subproblemas de tamaño más reducido, consiguiendo resultados similares (Aarts et al., 1997).

Es interesante establecer que las estructuras de información inicial para estos casos pueden ser alteradas para proporcionar la información a modo de árboles y no empleando la tradicional distancia euclídea o bien matrices de distancias, en cuyo caso sería mucho más lento el proceso de cálculo. Destacan las estructuras de información propuestas por Fredman et al. (1995), a modo de árbol, donde se describen una serie de reglas sencillas de interpretación para que el cálculo de distancias se realice mucho más rápido que recalculando todas las distancias euclídeas entre todos los nodos, o bien interpretando las distancias reales entre cada par de nodos.

En las siguiente sección analizamos con detenimiento una de las generalizaciones del problema del viajante o TSP, en concreto aquella que considera un número NV de vehículos en contra de un solo vehículo, y que se corresponde con el problema de planificación de rutas o VRP.

4 VRP: Vehicle Routing Problem

En esta sección se comentan cuáles son los principales fundamentos del problema de planificación de rutas de vehículos o VRP, así como de los diferentes métodos que han sido desarrollados para resolver el problema y que provienen, en su mayor parte, de los planteamientos recogidos para resolver el problema del TSP.

Para ello, se parte de la situación más sencilla descrita anteriormente. Se trata de una generalización del problema del viajante, en la cuál una flota de *NV* vehículos con capacidades limitadas, ha de servir un número *N* determinado de nodos en un tiempo determinado. Para ello la empresa dispone de un depósito o nodo central en el cuál están todos los vehículos preparados para servir los diferentes nodos. En este sentido cada vehículo ha de realizar una ruta, según la secuencia de nodos que ha de visitar. Estas rutas han de incurrir en la menor distancia total recorrida. Además existen restricciones de todo tipo, siendo las más comunes las referidas a las capacidades de cada vehículo y las distancias máximas recorridas en cada ruta.

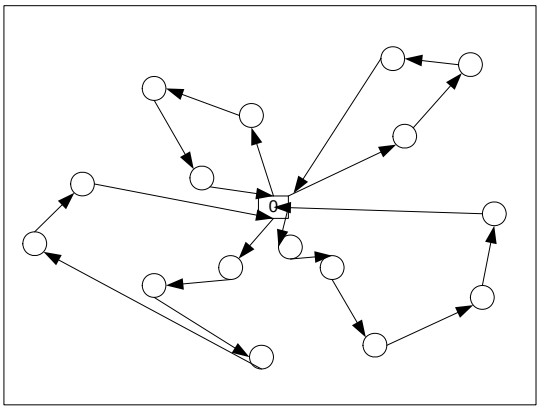


Figura 2.12 Planteamiento general del VRP.

Más formalmente definiríamos un conjunto *NV* de vehículos o viajantes, que deben de servir los *N* nodos procurando minimizar la distancia total recorrida desde un único depósito o almacén principal a los diferentes nodos donde hay que realizar un servicio. Cada vehículo *v* puede disponer de capacidades de carga diferentes, denotadas por K_v. Igualmente, y como ocurría en el caso del TSP, se dispone de información sobre los costes de desplazamiento entre cada par de nodos denotados por c_{ij} siendo los costes entre *i* y *j* simétricos. Además pueden existir restricciones temporales que obligan al vehículo a regresar al depósito central antes de un momento T_v, generalmente correspondiéndose con el cierre del depósito. De esta forma la solución consiste en obtener la matriz $X=[x_{ij}^v]$ donde x_{ij}^v es *1* si el vehículo *v* hace el trayecto de *i* a *j*, y *0* en cualquier otro caso. De esta manera la función objetivo consiste en minimizar los costes totales de transporte para la totalidad de las rutas. Formalmente podemos establecer lo siguiente:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{v=1}^{NV} c_{ij}\, x_{ij}^{v}$$

Sujeto a

$$\sum_{i=1}^{n}\sum_{v=1}^{NV} x_{ij}^{v} = 1 \qquad j = (2,..., n)$$

$$\sum_{j=1}^{n}\sum_{v=1}^{NV} x_{ij}^{v} = 1 \qquad i = (2,..., n)$$

$$\sum_{i=1}^{n} x_{ip}^{v} - \sum_{j=1}^{n} x_{pi}^{v} = 0 \quad (v = 1,..., NV;\ p = 1,..., n)$$

$$\sum_{i=}^{n} d_i \left(\sum_{j=1}^{n} x_{ij}^{v} \right) \le K_v \qquad (v = 1,..., NV)$$

$$\sum_{i=1}^{n} t_i^{v} \sum_{j=1}^{n} x_{ij}^{v} + \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}^{v} x_{ij}^{v} \le T_v \qquad (v = 1,..., NV)$$

$$\sum_{j=2}^{n} x_{1j}^{v} \le 1 \qquad (v = 1,..., NV)$$

$$\sum_{i=2}^{n} x_{i1}^{v} \le 1 \qquad (v = 1,..., NV)$$

$$x_{ij} \in S$$

$$x_{ij}^{v} = 0 \text{ ó } 1 \qquad \textit{para todo } i, j, v$$

Siendo:

N	Número de nodos
NV	Número de vehículos
K_v	Capacidad del vehículo v
T_v	Tiempo máximo permitido para la ruta del vehículo v
d_i	Demanda en el nodo i $(d_1 = 0)$
t_i^v	Tiempo necesario para que el vehículo v para enviar o recoger en el nodo i, $(t_1^v=0)$
t_{ij}^v	Tiempo necesario para que el vehículo v se desplace de i a j, $(t_{ii}^v= \infty)$
c_{ij}	Coste de viajar de i a j
x_{ij}^v	Igual a 1 si el vehículo v viaja de i a j y será 0 en cualquier otro caso
i, j, p	Nodos pertenecientes al conjunto N
S	Subconjunto de restricciones equivalentes a las formuladas para el TSP, y cuya finalidad es evitar subrutas

Como cabe esperar, la resolución directa de este planteamiento es una tarea ardua y muy compleja que se convierte en inviable para problemas de un elevado número de nodos (más de 150). Por ello es necesario recurrir a mecanismos de optimización que salven la aplicación directa de la programación matemática, ya sea a través de la relajación lagrangiana o bien a través del planteamiento de métodos heurísticos.

2.5 Problemas tipo de VRPs

Como problemas tipo para el caso del VRP existen multitud de bibliotecas disponibles a través de *Internet*. Las más utilizadas son las de Taillard[15] y las de Solomon[16]. Algunas de estas bibliotecas contemplan la existencia de ventanas de tiempo, y que en el caso del VRP son obviadas por los autores para implementar sus algoritmos sobre las baterías de problemas tipo.

Igualmente, las comparaciones en el caso del VRP no se pueden realizar con respecto a un único valor obtenido de manera exacta y próximo al óptimo como ocurría con el límite inferior de Held y Karp (1970) en el caso del TSP, sino que hay que hacerlo con los resultados mejores publicados hasta la fecha. Es cierto que dicho límite se puede calcular a través de la relajación lagrangiana del problema original, pero su cálculo no ha tenido tanta trascendencia como ocurre en el TSP. En este sentido se mantienen actualizadas bases de datos con los mejores resultados para gran cantidad de problemas tipo. Algunas de estas tablas y resultados se proporcionan en este trabajo.

A continuación se consideran los diferentes métodos propuestos por la literatura para la resolución de VRPs. Dentro de estos enfoques y al igual que ocurría en el caso del TSP, se desarrollan en primer lugar los enfoques exactos a través de la programación lineal o bien a través de la relajación lagrangiana. Se consideran también variantes de estos

[15] Disponibles en http://ina.eivd.ch/collaborateurs/etd/problemes.dir/vrp.dir/vrp.html
[16] Disponibles en http://www.idsia.ch/~luca/macs-vrptw/problems/

enfoques exactos. Con mayor profundidad se detallan los procedimientos de aproximación, o heurísticos, cuya aplicación es mucho más flexible y realista, dado que permiten una mayor inclusión de restricciones de manera que el problema se acerca mucho más a los casos reales que se plantean en las empresas.

2.6 Planteamientos exactos para la resolución de VRPs

Son modelos de optimización basados en la programación matemática para resolver el problema del VRP. En algunos casos se hace una programación matemática por pasos y se optimiza en dos fases. Se trata de algoritmos basados en programación dinámica, algoritmos de corte de planos y establecimiento de límites inferiores. Sin embargo, este tipo de métodos sólo son posibles para problemas con un número reducido de nodos. A continuación se consideran brevemente algunos de ellos.

2.6.1 Planteamientos basados en la relajación lagrangiana

Un ejemplo de estos algoritmos es el denominado *Algoritmo de Penalización* de Stewart y Golden (1979). Se trata de un algoritmo basado en la inclusión de la restricción de la capacidad de cada ruta en la función objetivo, aplicando un multiplicador indicativo de la penalización cuando la demanda de una ruta viola la restricción de capacidad del vehículo. Para ello se asume que la flota de vehículos es homogénea y que no hay restricciones temporales. Sin embargo se mantiene la diferenciación de cada vehículo *v* ya que cada uno realizará una ruta diferente, de manera que las secuencias de nodos que integrarán sus visitas son distintas, y por ello es necesario especificar qué visita se asigna a qué vehículo.

Igualmente se contempla un nuevo conjunto de restricciones S^* para hacer referencias al resto de las restricciones contempladas para el problema general del VRP y que pretenden eliminar las subrutas.

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{v=1}^{NV} c_{ij}x_{ij}^{v}$$

Sujeto a

$$\sum_{i}\sum_{j} d_i x_{ij}^{v} \le K \qquad v = 1,\ldots, NV$$

$$x_{ij}^{v} = 0 \text{ ó } 1 \qquad \textit{para todo } i, j, v$$

$$x_{ij}^{v} \in S^*$$

Además, podemos incluir la siguiente formulación adicional, para representar únicamente la asignación de la ruta más cargada al vehículo 1, la segunda más cargada al 2, y así sucesivamente.

$$\sum_i \sum_j d_i x_{ij}^k \geq \sum_i \sum_j d_i x_{ij}^l \qquad para\ l = k+1\ y\ k = 1,..., NV-1$$

De esta manera un posible planteamiento de la relajación lagrangiana sería:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{v=1}^{NV} c_{ij} x_{ij}^v \ + \lambda \sum_i \sum_j d_i x_{ij}^l$$

Sujeto a :

$$\sum_i \sum_j d_i x_{ij}^k \geq \sum_i \sum_j d_i x_{ij}^l \qquad para\ l = k+1\ y\ k = 1,..., NV-1$$

$$x_{ij}^v \in S^*$$

Donde λ se puede interpretar como una penalización por violar las restricciones de capacidad de las rutas y además $\lambda \geq 0$.

Mediante este planteamiento el algoritmo utilizado se desarrollaría de la siguiente manera:

1. Establecer $\lambda = 0$. Resolver el problema del VRP. Si la solución no viola las restricciones de capacidad, entonces se ha obtenido la solución final.
2. En caso contrario hacer $\lambda > 0$ y $\Delta > 0$, donde Δ es un valor incremental prefijado.
3. Resolver el problema. En este caso la ruta más cargada tiene una penalización de λ. Si la solución cumple las restantes condiciones, entonces ir al paso 5. Caso contrario seguir.
4. La ruta 1 está sobrecargada. Fijar un $\lambda=\lambda+\Delta$ y volver al paso 3.
5. Buscar el valor menor de λ, denotado por λ^*, entre $\lambda-\Delta$ y λ tal que la solución para ese valor λ^* cumpla todas las restricciones de S^*. Aportar la solución $X(\lambda^*)$.

De esta manera a través de estas iteraciones es posible encontrar una solución que cumpla con las restricciones a través de la reducción paulatina de las penalizaciones provocadas por las violaciones de carga de los vehículos.

Más recientemente, Mingozzi, Giorgi y Baldacci (1999) presentan un método exacto para el problema del VRP con retornos considerando tanto recogidas de clientes en determinados nodos como entregas a clientes en otros nodos. Para ello desarrollan un algoritmo de programación lineal, a través de su formulación exacta. Una vez conseguido esto, plantean la relajación de la lagrangiana del problema dual y a partir de esa relajación

desarrollan una serie de heurísticos para su resolución. En este caso se evitan afrontar el problema directamente y con el planteamiento exacto consiguen obtener los límites inferiores del coste mínimo total. Sin embargo este tipo de planteamientos exactos tienen el inconveniente de no ser válidos para problemas con un elevado número de nodos: generalmente son ineficientes para un número de nodos superior a 100.

2.6.2 Algoritmos de acotación y limitación (Branch and Bound)

Este tipo de algoritmos utilizan las restricciones enteras para acotar el espacio de la programación lineal en diferentes subproblemas, a través de la consideración de los rangos de variación de cada una de las variables. De esta manera se consigue una resolución del problema de programación lineal entera de una forma mucho más sencilla.

Resuelto un subproblema el método cambia a otro en el mismo nivel y repite el proceso. Si el subproblema no tiene solución, se acotará este problema y después continuará con los restantes subproblemas. A través de estas operaciones se obtendrá el valor de la solución óptima.

De esta manera a través de la acotación de cada uno de estos subproblemas o ramas se pueden obtener los límites inferiores de la solución óptima, que en todo caso se corresponderá con la combinación de valores de la solución que arroje el menor valor, y que será menor que el valor de la solución verdadera, por lo que se trata de un límite inferior a la solución del problema.

Una característica general de este tipo de algoritmos es que aunque generalmente permiten obtener el valor óptimo de la solución, la gran cantidad de cálculos necesarios lo hacen ineficientes para problemas de tamaño medio.

Mediante este planteamiento se han desarrollados otras variantes, como por ejemplo los algoritmos de corte de planos, que tienen un fundamento muy similar con la variante de que diseñan sus propias restricciones en cada paso.

Sin embargo dado que este trabajo persigue afrontar de manera eficaz y eficiente el problema de establecimiento de rutas, se han descartado los algoritmos exactos o basados en programación lineal debido a sus múltiples limitaciones. Por ello es necesario acudir a algoritmos de aproximación que sean capaces de proporcionar soluciones cercanas al óptimo.

Las características generales de estos dos tipos de planteamientos se basan en su gran complejidad derivada de la formulación matemática y de los tediosos cálculos necesarios para resolverla. En este sentido, si bien suponen un gran esfuerzo creativo e intelectual, no se tendrán en consideración dado que sus resultados no suponen grandes avances con respecto a los resultados conseguidos por los planteamientos heurísticos que en mayor medida, centran el estudio aquí planteado.

2.7 Planteamientos heurísticos del VRP

En los últimos años se han desarrollado multitud de técnicas para resolver el problema del VRP sin llegar a utilizar planteamientos de programación matemática. Entre estos métodos de podemos hacer una diferenciación en tres tipos, de manera análoga a la clasificación realizada para los métodos heurísticos aplicados al TSP.

- **Heurísticos de construcción de rutas**. Seleccionan sucesivamente nodos siguiendo un criterio especial hasta construir una solución posible o factible.
- **Heurísticos de mejora de rutas**. Parten de una solución factible y buscan una mejora que cumpliendo las restricciones (es decir siendo factible) mejora el anterior sistema de rutas.
- **Heurísticos en dos fases**. Realizan la asignación de nodos a cada vehículo en la primera fase y tratan de obtener la ruta óptima para cada vehículo en la segunda fase. Este tipo de soluciones se conocen también con el nombre de "cluster primero y ruta después", puesto que el primer paso del modelo es agrupar los nodos o ciudades en clusters, para posteriormente realizar la ruta en cada uno de los clusters correspondiéndose cada una con un vehículo diferente.

A partir de este momento centramos únicamente la atención en aquellos métodos para la resolución del problema VRP, con las siguientes particularidades:

- Flota homogénea
- Restricciones de carga en los vehículos
- Número limitado de vehículos
- Demandas variadas en los nodos
- Inexistencia de ventanas de tiempo, ni de restricciones temporales a las rutas
- Distancias euclídeas entre nodos
- Coste equivalente a la distancia total recorrida
- Función objetivo: Minimizar el coste total de las rutas.

2.7.1 Algoritmos de construcción de rutas

Los algoritmos de construcción de rutas parten de una situación en la que solamente se considera la dispersión geográfica de los nodos que han de ser servidos y las cantidades demandadas de estos nodos. Por ello el objetivo de estos planteamientos es construir una solución inicial de la mejor manera posible. Para ello se han desarrollado multitud de métodos, entre los cuáles destacan los siguientes.

4.1.1.1 Método de los ahorros

Se trata de conocido método de Clarke y Wright (1964), basado en un proceso de intercambio de nodos realizado por los ahorros producidos en la sustitución de los que en un momento dado están en la ruta por otros que aún no están. Para ello se calculan los ahorros con el siguiente planteamiento: si comenzamos con una situación en la cuál dos

vehículos deben de servir los puntos *i* y *j* y planteamos la situación en la que solamente un vehículo servirá ambos puntos, se obtendría un ahorro de :

$$(2c_{1i} + 2c_{1j}) - (c_{1i} + c_{1j} + c_{ij}) = c_{1i} + c_{1j} - c_{ij}$$

Después de calcular todos los ahorros para cada par de puntos, se ordenan de forma jerarquizada y una vez ordenados se escogerán los que mayor ahorro provocan, procediendo a su unión, sin violar las restricciones del modelo.

4.1.1.2 Métodos de inserción

Dentro del grupo de los métodos de inserción se recogerían todas las variantes para la construcción de rutas contempladas para el caso del TSP. Sin embargo, a la hora de aplicar estos métodos para la resolución del VRP, es necesario tener en cuenta alguna restricción adicional, que impone que se hayan de crear diferentes rutas para cada vehículo. Estas restricciones adicionales son las derivadas de un tiempo determinado que no ha de ser superado por ningún vehículo o bien una capacidad limitada de los mismos.

En general estos son los métodos más comunes para construir un conjunto de rutas iniciales, además de los clásicos del vecino más cercano.

La característica general de este tipo de métodos como ocurría en el caso del TSP es que se trata de métodos muy rápidos para encontrar una solución, que suele ser bastante mediocre y por ello susceptible de ser mejorada. Por ello se hace patente la necesidad de aplicar métodos de mejora de rutas, entre los que destacan los siguientes.

2.7.2 Algoritmos de mejora de rutas

Son similares a los aplicados en el caso del TSP y se basan en los planteamientos de Lin (1966). Estos métodos parten de una solución inicial factible y persiguen mejoras en función de intercambios de nodos en el orden que sigue la ruta, o bien entre rutas diferentes, tratando de aproximarse al óptimo.

Prosser y Shaw (1996) recogen en sus trabajos la aplicación de los cuatro heurísticos de mejora más aplicados en la literatura y establecen un método que contempla la aplicación simultánea de todos ellos en vez de aplicarlos por separado, enfoque tradicionalmente desarrollado por los demás autores.

Los heurísticos de mejora para el caso de los VRPs pueden ser tanto *intra-ruta* como *inter-ruta*. En concreto los intra-ruta serían los de tipo *k*-opt y más detalladamente, el de mayor aplicación para el caso de VRP que es el de tipo 2-opt. Dentro del grupo de los inter-ruta estarían los de tipo Reposicionado, Intercambio y Cruce.

El operador de cambio 2-opt es exactamente igual que el algoritmo 2-opt descrito para el caso de los TSP. Es decir se trata de establecer intercambios de arcos de unión de las

ciudades de una ruta determinada para establecer si de esa forma se mejora o no la distancia total. En caso afirmativo dicho cambio se realizará.

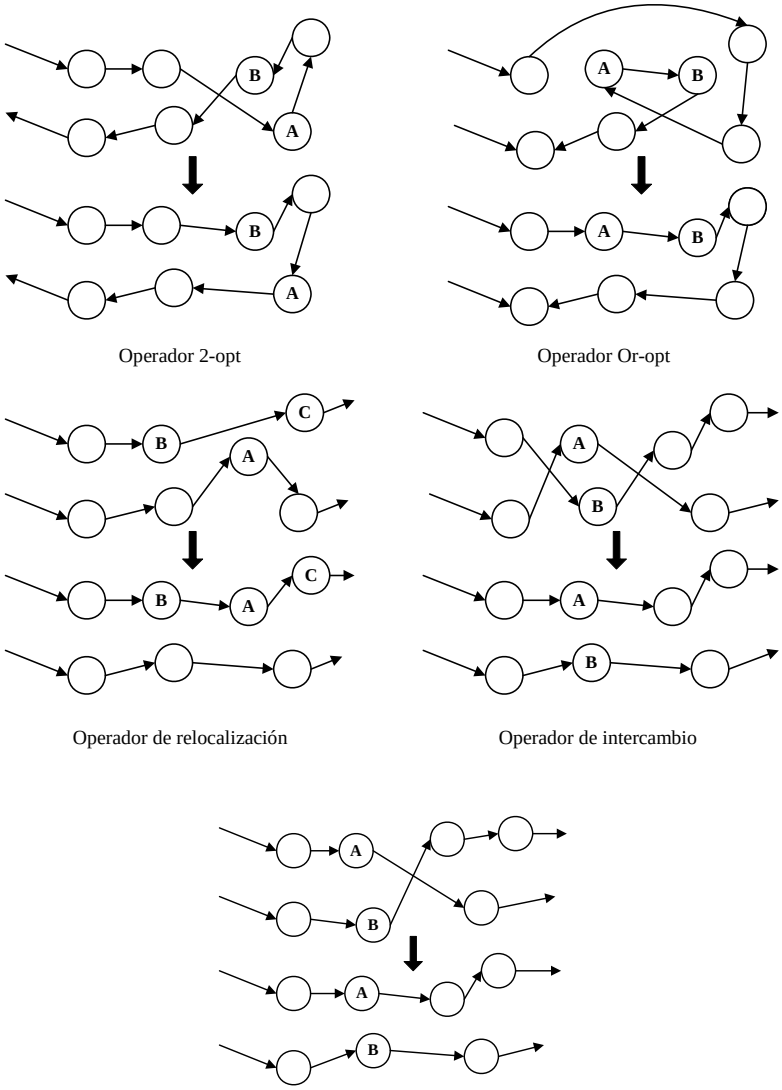


Figura 2.13 Ejemplos de operadores de mejora.

Dentro de este grupo estaría incluidos también los de tipo 3-opt y 4-opt o de doble puente. Sin embargo estos dos últimos no han tenido tanta aplicación, ya que la problemática característica de los VRPs y más en concreto la de los problemas estándar no exige tal complicación ya que si estamos tratando de realizar cambios en rutas preestablecidas, con un reducido número de nodos, casi nunca superior a 10 ó 15 nodos por ruta, no es necesaria tanta potencia de cálculo para realizar la secuencia de nodos óptima para una

ruta concreta. Es por ello por lo que la mayor parte de los métodos solamente contemplan el algoritmo 2-opt.

Sin embargo la peculiaridad del VRP sí exige procedimientos muy potentes que permitan la modificación de las rutas existentes para obtener el óptimo global y por ello se han desarrollado múltiples operadores heurísticos de mejora inter-ruta.

En concreto el operador de relocalización o de reposicionamiento de nodo implica que un nodo de una ruta se intercambie de una ruta a otra, es decir, reasignar una visita de un vehículo a otro, siempre que con ello se minimice la distancia total del problema. De la misma manera, un nodo, puede ser reasignado a una ruta vacía o nueva, resultando en la creación de un nuevo itinerario que ha de ser tenido en cuenta, para su posterior tratamiento, si esto es posible. Estas visitas que se desvinculan de una ruta existente frecuentemente se reasignan a un vehículo inexistente o virtual, para su posterior postoptimización.

El operador de intercambio consiste en que dos visitas de dos rutas diferentes se intercambien entre sí, por lo que las rutas mantienen el mismo número de visitas (algo que no ocurre con la reasignación vista en el apartado anterior), aunque con nodos diferentes a la situación inicial.

El operador de cruce intercambia los últimos arcos de dos rutas, es decir las cabeceras de dos rutas, generalmente adyacentes para reducir la distancia total del problema. También es posible intercambiar la cabecera de una ruta con el comienzo de otra resultando en la concatenación de rutas y el resultado de una ruta vacía. Es más, si una de las rutas del operador de cruce está vacía, entonces el resultado puede ser la división de la ruta que no está vacía en dos rutas no vacías.

Generalmente la aplicación de este tipo de modelos exige la existencia previa de rutas iniciales calculadas con algún heurístico de construcción. Normalmente se obtiene una serie de listas de visitas equivalentes a tantos otros vehículos de manera que a cada vehículo le corresponde una de las listas. Además es corriente la utilización de un vehículo virtual, encargado de hacer las visitas de las ciudades o nodos aún no asignados y que por lo tanto ha de ser cubierto por un vehículo inexistente. La lista de este vehículo ha de reasignarse utilizando alguno de estos heurísticos de mejora a las listas de los vehículos existentes y es aquí donde entran a jugar los heurísticos de mejora inter-rutas. Para ello se tienen en cuenta los costes de visita del vehículo virtual, inicialmente con capacidad infinita y que va a ser el que realiza la totalidad de las visitas, pero cuyos costes de visita son mayores que los del vehículo real, de forma que se fuerza a los algoritmos a que reasignen estas ciudades a los vehículos verdaderos, mediante los planteamientos expuestos anteriormente y cuando está envuelto el vehículo virtual entonces sólo es posible el heurístico de reasignación.

En este tipo de casos la función objetivo suele hacer referencia únicamente a la minimización de la distancia total recorrida, sin tener en cuenta otros posibles objetivos, como por ejemplo el aprovechamiento máximo de las capacidades de los vehículos, etc.

Uno de los últimos trabajos en los que se contempla la aplicación de heurísticos de mejora es el de Kilby et al. (1999). Estos autores consiguen mejorar mediante la aplicación conjunta de estos 4 métodos, 11 de los problemas de Solomon (1987) anteriormente resueltos por Taillard et al. (1995) y considerados como los mejores resultados hasta el momento. Aunque en estos dos casos, la aplicación de estos algoritmos sobre los problemas de Solomon (1987) se hizo contemplando las ventanas de tiempo existentes para todos los nodos, incluido el depósito. A estos métodos se presta mayor atención en el siguiente epígrafe sobre el VRPTW.

2.7.3 Algoritmos de dos fases: cluster primero, rutas después.

Se trata de algoritmos que en un primer momento persiguen obtener una repartición de los nodos que se han de servir en zonas o puntos de concentración, a las cuáles acudirán los vehículos de forma independiente y, en segundo lugar, para cada zona se obtiene la ruta o secuencia de puntos a servir. Algunos ejemplos de este tipo de algoritmos se desarrollan a continuación.

4.1.1.3 Heurístico de Gilett-Miller (1974)

El heurístico de Gilett-Miller (1974), también conocido como algoritmo *"sweep"* (barrido) se trata de un algoritmo que se desarrolla en dos fases. En primer lugar se asignan los nodos a cada vehículo y a continuación resuelve el problema del TSP para cada uno de los vehículos, estableciendo el orden en el que los puntos de demanda serán servidos.

Para resolver la primera fase es necesario contar con las coordenadas polares de cada uno de los puntos. Utilizando el depósito central como pivote inicial, se escoge un nodo semilla de forma aleatoria y a continuación se comienza a *barrer* en el sentido de las agujas del reloj o al contrario, de manera que se van agrupando los nodos a medida que se van barriendo, hasta cubrir la capacidad del vehículo. Matemáticamente es necesario calcular el ángulo formado por cada nodo con respecto al eje que une el depósito con el nodo semilla. A partir de ahí se escogerán los nodos que impliquen un menor ángulo hasta saturar la capacidad del vehículo y en ese momento se creará la primera ruta. Este proceso se repetirá hasta asignar todos los nodos a los vehículos.

La segunda fase se realiza resolviendo los TSPs para cada conjunto de nodos que deben ser visitados por cada vehículo. Para ello se puede aplicar cualquiera de los métodos descritos para la resolución de TSPs.

Este proceso se puede repetir con nodos semilla distintos para así encontrar la mejor solución.

2.7.3.1 Algoritmo de Fisher y Jaikumar (1981)

Se trata de uno de los algoritmos más recurrentes en el desarrollo de los métodos de cluster primero, rutas después. Este algoritmo resuelve en primera instancia el problema genérico de asignación de los clientes a una serie de nodos semilla previamente seleccionados. Para ello realiza las siguientes etapas en el cálculo de las rutas finales.

Fase 1 Selección de los clientes semilla.

En esta primera fase se selecciona un número de clientes semilla j equivalente el número de vehículos disponibles para realizar la planificación (*K*). Estos clientes semilla se denotan por j_k y generan los *K* diferentes clusters que posteriormente darán lugar a las diferentes rutas.

Fase 2 Cálculo de los costes de asignación de cada cliente *i* a cada cluster *k*

En esta fase se calculan los costes de asignación de cada cliente $i=1,...,N$ a cada uno de los clientes semilla j_k. Para ello se tiene en cuenta la distancia de cada nodo a cada cliente semilla a través del cómputo del coste asociado a dicha distancia y que se recoge en

$$d_{ij_k} = \min\left(c_{0i} + c_{ij_k} + c_{j_k 0}; c_{0 j_k} + c_{j_k i} + c_{i0}\right) - (c_{0 j_k} + c_{j_k 0})$$

Si suponemos que los costes de desplazamiento entre todos los nodos son simétricos, entonces la ecuación anterior resulta en:

$$d_{ij_k} = c_{j_k i} + c_{i0} - c_{j_k 0}$$

Fase 3 Resolver el Problema Genérico de Asignación

Tras obtener los costes de asignación de cada uno de los clientes a cada nodo semilla, entonces se procede a crear los cluster definitivos, y que en todo caso estarán limitados por la capacidad del vehículo que visite esa zona, y que se representa por *Q*. Para ello se contemplan las demandas de cada uno de los clientes, representadas por d_i.

Este problema se conoce con el nombre del Problema Genérico de Asignación (*Generic Assignment Problem*, o GAP)

En este caso el GAP consiste en asignar cada cliente a un vehículo *k* en cuestión, según sus limitaciones de capacidad. Para ello se implementa un problema de programación lineal entera en cada uno de los clusters donde la solución consistirá en los valores x_{ij_k} indicativos de si el cliente *i* se asigna al vehículo *k* o no, según tome valores de *1* ó *0*, respectivamente. Las restricciones que se imponen en este problema hacen referencia a que todo nodo ha de estar asignado a un cliente semilla en primer lugar, y que la suma de las demandas de los nodos asignados a ese cliente semilla no han de exceder la limitación del vehículo.

$$\min \sum_i \sum_j x_{ij_k} d_{ij_k} \qquad k = (1,...,K)$$

Sujeto a

$$\sum_j x_{ij_k} = 1$$

$$\sum_i d_i x_{ij_k} \leq Q$$

$$x_{ij_k} = (1,0)$$

Fase 4 Resolución del TSP en cada cluster

En esta fase se procede a la resolución de los TSPs de cada uno de los clusters a través de cualquiera de los procedimientos descritos en la sección anterior.

Este método permite la selección de diferentes grupos de clientes semilla proporcionando otras tantas posibles soluciones, de manera que su aplicación se puede realizar múltiples veces con la finalidad de obtener la mejor combinación de clientes semilla.

4.1.1.4 Heurístico de Bramel y Simchi-Levi (1993) y el algoritmo CCLP

De forma muy similar al caso anterior, Bramel y Simchi-Levi (1993) presentan un modelo general para la resolución del VRP basado en el problema de localización de puntos de concentración en función de capacidades (*Capacitated Concentrator Location Problem, CCLP*). En concreto aplican este modelo general a los dos problemas clásicos de VRP con capacidades limitadas y al VRP con Inventarios (*Capacited VRP* y *Inventory Routing Problem* respectivamente).

En el caso del CVRP, los autores se basan inicialmente en la solución de un problema de agrupación de demandas. Es decir, inicialmente los clientes se han de agrupar hasta conseguir lotes de igual tamaño a la carga del camión, esto es equivalente a saturar la ruta en función de la carga que ha de llevar.

Si tenemos en cuenta que el número de clientes aumenta, el tamaño medio de los lotes tiende a igualarse en torno a una constante determinada. Si además consideramos que los clientes se distribuyen de manera uniforme, en una región limitada y sus demandas se distribuyen de forma normal, siendo igualmente independientes, entonces, estadísticamente podemos obtener unos límites para la solución proporcionada por el heurístico. La construcción del límite superior es el fundamento del modelo desarrollado.

Se trata de un algoritmo asintóticamente óptimo, es decir que tiende al óptimo a medida que el número de clientes aumenta.

Una vez distribuidos los clientes, se sobrepone una cuadrícula en el plano. A continuación, en cada cuadro, se resuelve el problema de agrupación de clientes en función de sus demandas e igualando cada grupo a la carga de un camión. Para cada grupo de clientes se envía un camión. La ruta efectuada por el camión será un TSP entre los clientes asignados al camión y el depósito.

Para ello los autores proponen un mecanismo consistente en que la ruta comience en el depósito, vaya a un primer cliente (llamado cliente semilla) y, a partir de ahí, vaya y vuelva a cada uno de los clientes restantes de la ruta desde este cliente semilla y finalmente regresa al depósito.

Así el coste de cada ruta consiste en dos partes: el coste de ir y venir entre cada cliente y el cliente semilla y el coste entre el depósito y el cliente semilla. El objetivo es minimizar todos los costes para todas las rutas y de esta manera obtenemos la solución al problema de concentración. Es decir, que una vez planteado y resuelto este problema habremos obtenido tanto los clientes semilla (puntos de concentración) como las conexiones del resto de los clientes a cada cliente semilla, de manera que no se incumplan las restricciones de capacidad.

El siguiente paso consiste en convertir cada solución en una ruta para que el vehículo sólo pase una vez por cada cliente. Para ello tienen en cuenta los costes de inserción, buscando siempre la inserción más barata para decidir en qué lugar de la ruta se incluye al cliente. Así convierte la solución en forma de estrella en una ruta verdadera.

4.1.1.5 Heurístico de Bramel y Simchi-Levi (1995) y el algoritmo LBH

De forma similar al caso anterior Bramel y Simchi-Levi (1995) proponen este método basado en el establecimiento de clusters de clientes para ser integrados en cada una de las rutas (*Location Based Heuristic*, o *LBH*). Para ello proponen el siguiente algoritmo basado en tres fases diferenciadas:

- **Fase 1:** se escogen *m* subconjuntos de *N*, siendo *N* el número total de nodos. De esta manera se crean diferentes subconjuntos de clientes servidos por cada vehículo de una manera no necesariamente posible. Además, estos subconjuntos pueden solaparse e incluso pueden no llegar a cubrir el total de clientes existentes. A continuación se calculan los costes de servir a cada uno de estos subconjuntos, para lo cual se establece una base y se calculan los costes de acudir desde el depósito central a cada una de las bases, al igual que ocurría en el caso de los puntos de concentración del CCLP (*Capacited Concentrator Location Problem*). Además se calculan los costes de servir a cada uno de los clientes dentro de cada uno de los subconjuntos creados.
- **Fase 2:** se resuelve el CCLP con los datos proporcionados en la primera fase. Este problema consiste en escoger algunos de estos conjuntos de bases y los

costes de conexión de cada nodo a una de estas bases, de manera que el coste total de establecer dichas bases más la suma de los costes de conexión sea lo menor posible. La solución del problema de CCLP se puede obtener a través del planteamiento de la lagrangiana para el problema tipo de programación lineal entera.

- **Fase 3:** convertir la solución del CCLP en una solución del problema de planificación de rutas, con lo que se especifica qué clientes están conectados a qué bases o terminales. En este sentido se ha creado una partición de los clientes de manera conjunta por los diferentes vehículos desde el depósito central.

En este planteamiento del LBH se pueden presentar dos alternativas teniendo en cuenta la forma de considerar los costes de conexión.

Podemos hablar de costes de conexión basados en los costes de inserción, de manera que el coste total de conexión de todos los nodos de un conjunto se obtiene a través del establecimiento de los costes de inserción de cada nodo en una ruta que parte del nodo base, o terminal. En este caso estaríamos hablando de "*Seed Tour Heuristics*". También podemos hablar de los costes de conexión directos entre cada uno de los clientes y el nodo base, con lo que tenemos una forma de estrella entre todas las uniones de un mismo conjunto. En este caso estaríamos hablando de "*Star Connection Heuristics*".

2.7.4 Procedimientos compuestos: Metaheurísticos

En este apartado se muestran los procedimientos compuestos desarrollados en la literatura para la resolución del VRP. Al igual que en el caso del TSP, los algoritmos heurísticos iniciales, tanto de construcción, como de mejora, expuestos anteriormente, pecan de ser algoritmos deterministas que cuando entran en una zona de iteraciones que provocan descensos en la función objetivo, simplemente terminan con una solución de óptimo local, muy distante en ocasiones del óptimo global. Por esto al igual que ocurría en el planteamiento de los TSPs, se han propuesto nuevos métodos de búsqueda para provocar que los algoritmos existentes huyesen de estas soluciones locales, provocando la búsqueda en otras zonas del espacio de soluciones.

4.1.1.6 Búsqueda Tabú

En el caso del VRP también se han desarrollado múltiples variantes de la búsqueda Tabú, siguiendo los mismos fundamentos que motivaron su aplicación para el TSP. Algunos de estos planteamientos de búsqueda tabú se describen seguidamente.

Algoritmo de flor de Rego (1998)

Tienen su origen en los algoritmos basados en el grupo de los *Path ejection algorithms*, consistentes en la segregación de una serie de nodos de la ruta para considerar a continuación otras alternativas o uniones. Son algoritmos similares a los del tipo de 2-opt y 3-opt, salvo que en este caso existe una figura de referencia que se utiliza de patrón. En

concreto, en este caso la figura de referencia para el caso del VRP y dada su similitud, es una flor, con tallo y pétalos.

El origen de este algoritmo lo desarrolla Rego (1997) en un trabajo previo donde plantea un algoritmo para la resolución de TSPs basado en una figura de ciclo unido a un tronco, inspirado en los desarrollos teóricos de Glover (1992) y en la teoría de grafos para la resolución de problemas de tipo combinatorio (Berge, 1970)

En este caso se trata de un nuevo algoritmo desarrollado por Rego (1998), para resolver problemas de VRP con limitaciones de carga y de distancia, incluyendo distancias simétricas. Se aplica al problema genérico de VRP. En este algoritmo, basado en la optimización local, se plantea el intercambio de nodos en la secuencia de visitas de una ruta o bien de rutas diferentes.

Es un método que recuerda a los clásicos 2-opt y 3-opt, e incluso el algoritmo de Lin-Kernighan (1973), dado que se fundamenta en cambios en las permutaciones de nodos visitados. Para ello el autor parte de una situación inicial compuesta por las rutas, identificadas con los pétalos de una flor. El siguiente paso es desligar una de estas rutas, o pétalos, para que se convierta en el tallo y de esta forma jugar con las diferentes alternativas de unión del tallo para volver a convertirlo en un nuevo pétalo.

El mecanismo de búsqueda tabú se aplica a dos niveles diferentes para obligar al propio algoritmo a que busque soluciones no exploradas con anterioridad. Para ello se penalizan los movimientos repetidos a la hora de deshacer los pétalos, para que busque nuevas segregaciones de rutas.

Se trata de un algoritmo basado en un concepto eminentemente gráfico y que, dado su enfoque de búsqueda de óptimos locales, pierde visión a la hora de afrontar la búsqueda global, aún a pesar de contener fuertes penalizaciones para los movimientos tabú. Por término medio se obtienen desviaciones del 2.5% sobre las mejores soluciones del momento. Sin embargo sí reporta ventajas en cuanto al tiempo de cálculo ya que al no contener cálculos complejos, la velocidad a la hora de obtener la solución indicada es mayor que otros algoritmos que consiguen mejores resultados. En concreto se establece una comparación con el algoritmo LBH de Simchi-Levi (1995), al igual que el de barrido mejorado de Renaud, Boctor y Laporte (1996). El LBH resulta ser uno de los algoritmos que más tiempo consumen a la hora de aportar una solución y además suele desviarse mucho con respecto a las soluciones optimas publicadas sobre los problemas tipo (alrededor del 5.27%), comparado con el 2% del algoritmo de la flor. Por su parte, el barrido de Renaud et al. (1996), consigue resultados más cercanos al óptimo que el LBH (desviación media de 2.43%), pero aún algo distantes del algoritmo de la flor (desviación media de 1.54%) y además el consumo de tiempo es mayor.

Otro de los ejemplos de búsqueda Tabú es el algoritmo de Kelly y Xu (1997) denominado *Set-Partitioning-Based Heuristic*. Contiene dos fases bien diferenciadas. La primera fase consiste en la generación de rutas, basada en los heurísticos tradicionales,

incluyendo algún mecanismo de mejora y la segunda fase consiste en procedimientos de ensamblado de estas rutas para construir la solución final. Es en estos procedimientos de ensamblado donde se aplica la búsqueda tabú.

El objetivo de la primera fase en construir un número elevado de rutas fijado previamente, pues se fundamentan en la hipótesis de que aún siendo soluciones pobres, sí van a contener determinadas uniones que posteriormente podrán ser reintegradas en la solución final. Para ello se generan multitud de soluciones o rutas alternativas obtenidas con cualquier método (datos históricos, computación paralela, etc.). En concreto los autores escogen dos métodos: el método de los ahorros ponderados de Paessens (1988) y el heurístico de construcción propio, fundamentado en la eyección de cadenas de nodos.

El método de los ahorros ponderados de Paessens (1988), simplemente incluye como variación del método de Clarke y Wright (1964) la consideración de dos parámetros *f* y *g*:

$$s_{ij} = d_{0i} + d_{0j} - g \cdot d_{ij} + f \cdot \left| d_{0i} - d_{0j} \right|$$

$$\text{dónde} \quad 0 < g \leq 3 \ \text{ y } \ 0 \leq f \leq 1$$

Aplican este método probando con diferentes valores de *f* y *g* de manera aleatoria y, a continuación, aplican un mecanismo de mejora de 3-opt a los TSPs de cada ruta generada.

El heurístico de construcción que presentan los autores se basa en dos posibles movimientos, el "*ejection-chain based move*" similar al propuesto por Rego (1997) y un movimiento de intercambio de dos clientes entre dos rutas diferentes. Estos movimientos se recogen en la figura 2.14. Para facilitar ambos movimientos se relajan las restricciones de capacidad incluyendo una penalización por sobrepasar dichas capacidades. Esta penalización se calcula de forma automática por el algoritmo, de manera que se van incorporando modificaciones atendiendo a estas sobrecargas.

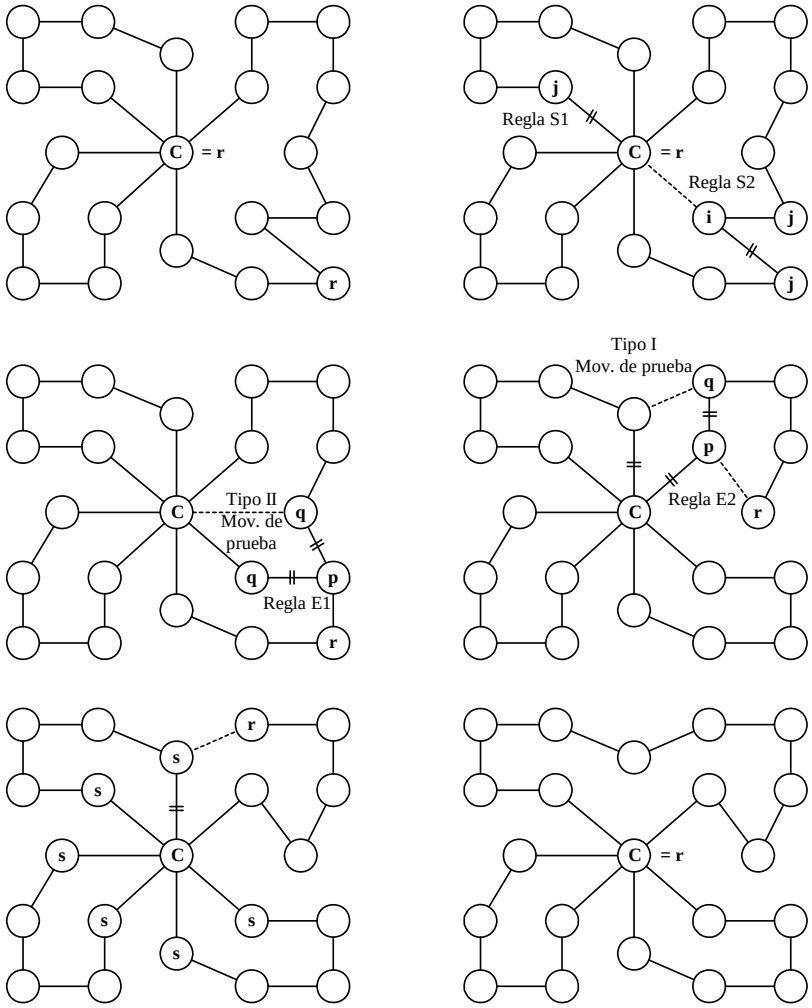


Figura 2.14 Desarrollo de dos movimientos alternativos para la modificación de las rutas, inspirados en el planteamiento del algoritmo de Lin-Kerninghan (1973).

En la fase de integración de las rutas en la solución final, se aplica un modelo de creación de conjuntos de nodos y se integran todas las rutas consideradas como mejores. Esta integración se puede realizar con programación lineal entera o bien aplicando nuevamente procedimientos heurísticos. La programación tiene la ventaja de resultar mucho más sencilla que la programación inicial del problema y por ello más fácil de resolver. Sin embargo los autores desarrollan un nuevo heurístico para resolver la elección de rutas finales.

El primer paso es escoger del total de rutas generadas, aquellas que más clientes cubren, manteniendo las restricciones de capacidad y longitud máxima permitida. De esta manera se seleccionan tantas rutas como vehículos. En este momento habrá clientes que estén

servidos múltiples veces y otros que no estén servidos por lo que será necesario corregir estos errores. Para ello se definen dos procedimientos adicionales:

- Eliminación de clientes repetidos. Se calculan los ahorros provocados por la eliminación del cliente en todas las rutas en las que esté representado y se elimina de todas salvo de la que provoca un ahorro máximo.
- Inserción de clientes no cubiertos. Se insertarán en las rutas que provoquen menor incremento de coste, manteniendo las restricciones de capacidad y longitud.

Además se intentará que con estos movimientos no se repitan rutas ya existentes. Una vez cerrado el ciclo de readaptación de rutas, se añaden éstas al número inicial N, obteniendo un número final de $N+N$ rutas.

A continuación se ordena la lista de rutas en función de su capacidad utilizada y, a partir de ahí, se utiliza esta lista para escoger aquellas rutas que estarán en la solución final. Para ello, se agrupan las rutas en función de su similitud y entre aquellas que no son similares (una de cada grupo) se plantean movimientos de permuta de clientes.

El algoritmo intenta penalizar aquellas rutas o regiones que están sobre-estudiadas, mientras que trata de primar la búsqueda o inclusión de rutas que se infrautilizan. De esta manera se hace mayor énfasis bien en la intensificación o bien en la diversificación de la búsqueda. Esto permite la parametrización del problema y es lo que los autores denominan "*oscilación estratégica*" y que hace referencia a la aplicación de la búsqueda tabú para que el modelo intente localizar nuevas áreas para la exploración.

Una vez escogidas las rutas definitivas, se plantea un mecanismo de mejora a través de los métodos heurísticos de mejora del TSP. En concreto, el 3-opt y la búsqueda tabú en TSP (TSTSP), penalizando movimientos para que no se deshagan hasta pasado un determinado número de iteraciones.

4.1.1.7 Recocido Simulado

La aplicación del SA para el caso del VRP también ha tenido relevantes exponentes, como es el caso de los trabajos de Kirkpatrick et al. (1983). En la resolución de los TSP se muestra como una técnica que proporciona buenos resultados dado que la aleatoriedad en la implementación de los intercambios de arcos generalmente permitía obtener buenas soluciones. Recordamos que se trata de un algoritmo asintóticamente óptimo. Sin embargo el vecindario de soluciones en el caso del VRP, para mecanismos de intercambios de 2, 3 o k arcos simultáneamente es mucho más reducido, dado el pequeño tamaño de las rutas generadas.

Por ello, la aplicación de la búsqueda aleatoria de intercambios de arcos dentro de las rutas, ha sido bastante limitada en el caso del VRP. Sin embargo se han desarrollado algunos intentos de aplicar este planteamiento, a través de los métodos de tipo 2-opt, o 3-opt e incluso recombinación de nodos en diferentes rutas con mayor o menor éxito. Si

comparamos los resultados obtenidos por estos algoritmos con los resultados proporcionados por la búsqueda tabú o los algoritmos genéticos, descartamos la aplicación directa de este planteamiento, dado sus mediocres resultados y los interminables tiempos de computación necesarios.

A pesar de ello y siguiendo los fundamentos del modelo, se está investigando la forma de aprovechar la información acumulada de las soluciones buenas encontradas en cada iteración, tal y como se planteaba en el algoritmo "*Co-Operative Simulated Annealing*", descrito en la sección anterior. En todo caso, esta línea de investigación está en un estado muy incipiente y por lo tanto susceptible de nuevas mejoras.

4.1.1.8 Algoritmos Genéticos

Al igual que ocurre con la búsqueda tabú, la aplicación de los algoritmos genéticos para el caso del VRP no se ha hecho esperar y de forma menos tímida que cuando ocurrió para el TSP. Multitud de planteamientos han surgido para adaptar los algoritmos genéticos a la resolución de los problemas de VRP. Duncan (1995) aporta los primeros estudios experimentales sobre diferentes casos recogidos en la literatura, en los cuales prueba la utilidad de diferentes algoritmos. En mayor profundidad estudia cómo evoluciona la aplicación de las mejoras generacionales empleando distintos heurísticos. Son interesantes las conclusiones que obtiene de sus estudios.

- Los cambios en el algoritmo de barrido no son tan efectivos como los intercambios de arcos (*k*-opt)
- Mantener la mejor solución encontrada y trabajar sobre ella es una buena idea
- La exploración sistemática del vecindario es efectiva, particularmente durante la fase de intensificación.
- Es interesante partir de soluciones buenas como solución de partida, ya que se obtienen mejores resultados y además acelera el proceso.
- No hay ninguna ventaja derivada de la sustitución incremental en contra de la sustitución generacional.

En la sección anterior se revisaba la aplicación de estos procedimientos para el caso del TSP, centrando sobre todo la atención en los diferentes mecanismos de combinación que se habían diseñado para obtener nuevas generaciones de soluciones. El fundamento para el caso del VRP es exactamente igual, dado que se plantean el mismo tipo de cambios sobre soluciones generadas para obtener así nuevas soluciones. Lo que es diferente son los mecanismos utilizados para realizar tanto las mutaciones como las recombinaciones. En este apartado se contemplan algunos de los métodos existentes.

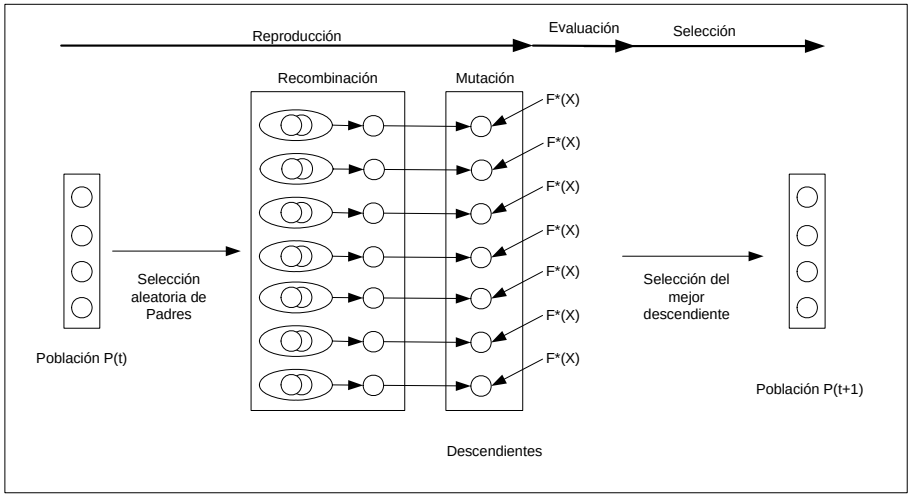


Figura 2.15 Planteamiento general mediante la teoría evolutiva de los algoritmos genéticos. Homberger y Gehring (1999)

Operador de cruce basado en la secuencia de nodos (*Sequence Based Crossover*)

Una vez seleccionados dos padres como dos soluciones individuales, con algunas semejanzas entre ellas, se planean los siguientes cambios para obtener la siguiente generación:

- Se selecciona de forma aleatoria una unión en cada uno de los padres. A continuación los clientes servidos antes de la ruptura del primer padre se unen a los clientes servidos después de la ruptura en el segundo padre. De esta manera se crea una nueva ruta en la solución.
- Un nuevo hijo se puede crear si se invierte el proceso desarrollado anteriormente.

Generalmente la solución proporcionada ha de ser corregida, dado que por un lado existirán nodos que hayan quedado descolgados y será necesario reintegrarlos y por otro lado habrá nodos que estén duplicados en más de una ruta, por lo que habrá que reorganizarlos. Para ello se pueden aplicar los algoritmos descritos en la sección anterior de mejora de rutas.

Operador de cruce basado en las rutas (*Route Based Crossover*)

En este caso el algoritmo de recombinación que se propone consiste en el reemplazo de una de las rutas de un padre, por otra de las rutas del segundo padre, creando un hijo con caracteres de ambos completamente diferente. Al igual que en el caso anterior, se crearán duplicidades entre las ciudades servidas por más de una ruta e incluso ciudades o nodos que han quedado sin servicio, por lo que habrá que aplicar mecanismos de corrección como en el caso anterior.

Mecanismos de Mutación.

Al igual que ocurría en el caso del TSP, se plantea la necesidad de establecer mecanismos que permitan establecer mutaciones o cambios internos en las soluciones generadas. El esquema general es el planteado seguidamente.

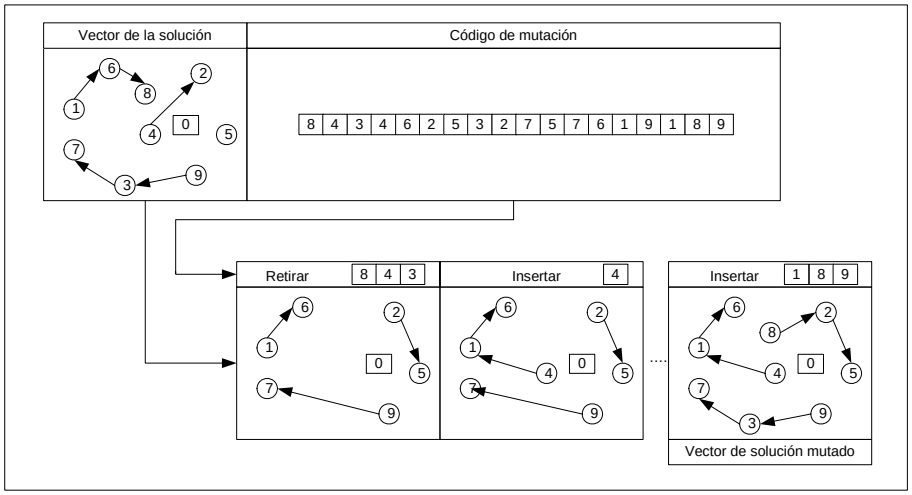


Figura 2.16 Ejemplo de una mutación sobre una solución individual Homberger y Gehring (1999)

Estos mecanismos de mutación se pueden considerar como una evolución de los planteamientos de intercambios inter e intra ruta, descritos anteriormente dentro del conjunto de mecanismos para mejora de rutas.

En general este tipo de planteamientos consiguen los mejores resultados dentro del conjunto de los heurísticos para la resolución de VRPs, tanto los desarrollos derivados de lo búsqueda tabú y el caso del algoritmo de Xu y Kelly (1999) como a través de los distintos planteamientos de los algoritmos genéticos. Actualmente se plantean diferentes combinaciones de todos los métodos resultando difícil su clasificación dentro de uno u otro grupo. En los últimos años y debido a las mejoras conseguidas en las aplicaciones de programación matemática, los planteamientos exactos y los de la relajación de restricciones empiezan a cobrar fuerza hasta el punto de competir con los mejores planteamientos heurísticos. A pesar de ello, estos nuevos enfoques de la relajación a través de la lagrangiana exigen disponer de software específico para la resolución de problemas, algo que dificulta su aplicación "*stand alone*" para la resolución de los problemas empresariales. Algunos de estos casos se comentan a continuación.

2.8 Búsqueda local con programación de restricciones

En este caso se describen los trabajos de Kilby, Prosser y Shaw (1999) quienes desarrollan un algoritmo de resolución fundamentado en la programación de las restricciones a ser aplicadas sobre una solución inicial de rutas previamente generadas. En concreto parten de tres algoritmos de construcción de rutas: el algoritmo de los ahorros,

los de inserción y un combinado entre ahorros e inserción que denominan *savings-insertion*. Sobre las soluciones de este último algoritmo plantean la aplicación de las restricciones que generalmente se producen en la vida real, en concreto restricciones temporales, de capacidad y de secuencia de visitas, de forma que flexibilizan mucho el modelo inicial, ya que permite adaptarlo a casos reales. En este sentido ya entran dentro del grupo de los planteamientos del VRPTW que se recogen en el siguiente apartado.

El planteamiento de estas restricciones se realiza empleando el software *ILOG Solver* (versión 4.3) y la librería *Distpacher*, software comercial de planteamiento de restricciones para problemas de VRP. Los programas se realizan en C++. La forma de operar de su modelo consiste en mejorar la ruta por medio de movimientos de búsqueda local. Para cada movimiento planteado se realiza un estudio de las restricciones desarrolladas. Si éstas son violadas, entonces el movimiento se declara como ilegal, en caso contrario el movimiento es legal y podrá ser realizado, siempre que mejore la distancia total de la ruta.

En cuanto a los heurísticos de mejora que emplean para perfeccionar las rutas obtenidas a través de su construcción inicial, hace referencia a dos métodos.

- **Tradicional**: utilizan el conjunto de técnicas de mejora de rutas tradicional y aplican el método de *Steepest Descent* para obtener soluciones más cercanas al óptimo pero que no dejan de ser soluciones de óptimo local. Rechazan todos aquellos movimientos que a pesar de reducir la distancia total del problema en cuestión, supongan la violación de alguna de las restricciones planteadas por el modelo.
- **Dirigida a las restricciones**: Dada una solución, se extrae una parte de esa solución y se reinserta en la solución parcial utilizando un proceso de búsqueda cuasi-completo. En este proceso, si la reinserción proporciona una distancia mejor, entonces se guarda, pero no se realiza.

Este proceso se repite hasta que se da una cierta condición preestablecida. Se trata de un proceso de búsqueda local puesto que se examinan todas las posibles alternativas para mejorar una solución determinada. Sin embargo, dado que el conjunto de posibles aplicaciones de esta búsqueda local es tan amplio, es necesario establecer algún mecanismo de reducción. Los autores definen este proceso como *Large Neighbourhood Search* (LNS) dadas sus características de búsqueda tan amplia.

Una vez planteadas las restricciones, concluyen afirmando que el mejor algoritmo es el de inserción dado que su búsqueda es más miope que la del algoritmo de los ahorros y que por lo tanto asume de manera más sencilla y fácil las restricciones impuestas que el algoritmo combinado de ahorro–inserción.

Partiendo de la aplicación de un método de construcciones de rutas se consiguen resultados entorno al 15% y 20% peores que los mejores publicados. Sin embargo esto se reduce a un 2% ó 4% a través de los procedimientos de mejora de búsqueda local.

Esta línea de investigación persigue adaptarse a situaciones reales donde existen multitud de restricciones que han de ser modelizadas de alguna manera. Así, por ejemplo, es posible utilizar restricciones temporales, ventanas de tiempo, restricciones de capacidad, restricciones de descansos obligatorios para los conductores, restricciones de tiempos máximos de rutas, etc.

Estos modelos exigen la utilización de software de desarrollo de restricciones y sus resultados, aunque generalmente peores en cuanto a los costes obtenidos comparados con los procedimientos o heurísticos clásicos, sí son mucho más reales y proporcionan una planificación de rutas asumible por una empresa.

La ventaja de los programas de desarrollo de restricciones es que basta con definir las variables de decisión para que el algoritmo implemente las restricciones. De esta manera a cada variable definida se le asignan un rango de valores posibles y el programa asume la restricción derivada sin necesidad de desarrollar matemáticamente todas las restricciones de manera formal.

Sin embargo este enfoque plantea también ciertos inconvenientes, generalmente derivados de que la comprobación de todas las restricciones para cada movimiento planteado exige mucho tiempo, por lo que es generalmente inviable la aplicación en su versión completa. Por este motivo se han desarrollado técnicas para acelerar la implementación de la Programación Restringida. Por ejemplo, aplicando únicamente las restricciones con mayor probabilidad de ser infringidas, dejando las demás para la comprobación de la solución final. Además es necesario tener en cuenta que solamente se aplicará el contraste con las restricciones si el movimiento efectuado reduce la distancia total, caso contrario se desecha.

Es interesante asociar el método de búsqueda perseguido en este tipo de modelos con la programación dinámica. El modelo comienza con una solución consistente en que cada nodo va a ser visitado por un único vehículo virtual y el objetivo es minimizar la distancia total. A partir de esa estructura de estrella se comenzarán a unir nodos mediante algún planteamiento de búsqueda, comenzando por el método de los ahorros, heurísticos de inserción y métodos de búsqueda local. A pesar de ello, la función objetivo sólo incluye la minimización de la distancia total.

Este tipo de enfoque suele conseguir por término medio resultados que oscilan entre un 3 y un 5% sobre los costes de los mejores resultados publicados. Sin embargo reportan una gran ventaja en cuanto al tiempo de computación sobre todo si se restringen los cálculos necesarios. De esta manera para resolver un problema de entre 100 y 200 nodos en un portátil con un procesador *Pentium* a 120Mh., solamente son necesarios unos 20 segundos, utilizando el software de *ILOG*. Esto es debido a que no se utiliza ningún tipo de metaheurístico para la resolución sino simplemente búsqueda local restringida. Algunos de estos resultados se pueden observar en De Backer, Furnon, Kilby, Proesser y Shaw (1999).

2.9 Optimización interactiva

Por último se contempla una nueva línea de resolución de los problemas de VRP, fundamentada en lo que se ha denominado Optimización interactiva. Este tipo de soluciones busca interactuar con el usuario del modelo de manera que se permite una mayor flexibilidad. Se trata por lo tanto de parametrizar el problema, establecer variables que el usuario puede controlar y alterar para encontrar la solución que más se aproxime al óptimo. Además también permiten el establecimiento de criterios completamente subjetivos, que por razones ajenas a la optimización han de acontecer de aquella manera (por ejemplo, pares de nodos que han de ser servidos secuencialmente). Obviamente, la mayor aceptabilidad de los métodos desarrollados por la literatura, a la hora de ser implantados en situaciones reales, depende de la flexibilidad de los mismos para ser adaptados a cada caso concreto, por lo que es necesario dotarlos de estructuras que permitan en algunos casos mayor flexibilidad.

Un ejemplo de este tipo de implementaciones lo encontramos en Duchessi, Belardo y Seagle (1988) quienes realzan las posibilidades de los sistemas expertos en la resolución de diversos problemas en Investigación Operativa. Inicialmente, se construye un DSS (*Decision Support System*) para el VRP. Posteriormente, comprueban cómo añadiendo a dicho modelo un sistema experto basado en el conocimiento autónomo, el modelo se convierte en una potente herramienta de decisión. Efectúa una integración de sistemas expertos basándonos en el conocimiento y en sistemas de soporte a la dirección. Utiliza en gran manera el conocimiento del operador, interrelacionándolo con el programa desarrollado.

Esta interactuación es una tendencia que se está imponiendo en todos los desarrollos, y que mayoritariamente se corresponde con las últimas etapas de la planificación de las rutas, en las que el usuario ya dispone del conjunto de rutas arrojado por el algoritmo, y sobre esa planificación plantea una serie de cambios con la intención de postoptimizar, o bien de implementar algunos cambios o modificaciones de última hora. Igualmente se trata de una vía para realizar los análisis de sensibilidad de los parámetros y valores del problema inicial, y que permiten al usuario conocer la variabilidad de la solución ante estas modificaciones previstas.

Hasta aquí se han desarrollado las diferentes líneas de resolución del problema del VRP, o planificación de rutas de tranpsorte. En la siguiente sección se recogen de forma más detallada los planteamientos referentes al problema del VRPTW, como caso particular del VRP en el que se integran plazos de servicio en cada nodo.

5 VRPTW: Vehicle Routing Problem with Time Windows

En este apartado se centra la atención en el estudio el problema más conocido de Planificación de Rutas con Ventanas de Tiempo (o VRPTW). Se presenta en esta sección una descripción de los métodos más utilizados por los autores en la literatura para afrontar la solución de este tipo de problemas, que cada vez son más comunes en el mundo real tanto en el transporte de viajeros, como en el de mercancías. El VRPTW consiste en la incorporación al problema genérico de VRP de las ventanas de tiempo, fundamentalmente a través de la introducción de tres variables adicionales en el planteamiento general que son: el momento de apertura del cliente en cuestión, el momento de cierre de las instalaciones de dicho cliente, y el tiempo de servicio de un vehículo a ese cliente. Igualmente se incorporan los momentos de apertura y cierre del depósito central.

El VRPTW consiste en una problemática relativa al diseño de rutas con el menor coste posible desde un único depósito hacia un conjunto de clientes geográficamente dispersos. Las rutas se han de diseñar de tal manera que respeten las restricciones de carga máxima de cada vehículo, los momentos de apertura y cierre de cada uno de los clientes, los tiempos de servicio a cada uno de estos clientes, así como el número total de vehículos, y los momentos de apertura y cierre del depósito central. Las aplicaciones de este tipo de problemas en la realidad son múltiples y afectan a diferentes situaciones cotidianas tales como: envíos bancarios, envíos postales y de paquetería, recogida de residuos urbanos y/o industriales, planificación de rutas para autobuses escolares, servicios de vigilancia privados, etc.

Ante la inclusión de tal cantidad de variables y restricciones en los planteamientos generales del problema es necesario aplicar técnicas de resolución heurísticas, dado que es ineficiente aplicar técnicas de resolución exactas para problemas con un número considerable de clientes. De esta manera se presentan en esta sección las dos líneas de investigación que se han abierto para resolver este problema de forma cuasi-óptima.

- Por un lado nos encontramos con los algoritmos de construcción de rutas, que pretenden aportar una primera solución del problema aplicando determinadas reglas más o menos intuitivas de decisión, y cuyos comienzos son los algoritmos de los ahorros de Clarke y Wright de 1964, aunque el primero en plantear una batería de problemas y los primeros métodos de solución de este tipo de problemas es Solomon en 1983.
- En segundo lugar nos encontramos con los procedimientos de mejora de rutas, también conocidos por Metaheurísticos, y que conforman un grupo de métodos de solución fundamentados en técnicas eminentemente matemáticas, generalmente de búsqueda intensiva.

Los problemas relativos al VRPTW han estado sujetos a múltiples investigaciones desde 1985. Las primeras investigaciones sobre este tipo de problemas, y sus métodos de

solución se pueden encontrar en Golden y Assad (1986), Desrochers et al. (1988), Golden y Assad (1988), y Solomon y Desrosiers (1988). Si bien en los primeros momentos se abordaron las posibles soluciones siempre a través de la aplicación de las técnicas heurísticas, en los últimos años se han desarrollado nuevas líneas de investigación a través de la aplicación de técnicas de resolución exactas como Desrosiers et al. (1995) y Cordeau et al. (2001), siendo uno de los mejores trabajos de recopilación de estas técnicas es la Tesis Doctoral de Larsen (1999).

En el trabajo del SINTEF en el que se basa esta sección, los autores Bräysy y Gendreau (2001) concluyen que se ha realizado poco trabajo de investigación y comparación de los métodos heurísticos, y que al mismo tiempo éstos son de gran importancia, tanto por los avances que individualmente suponen cada uno de ellos, como por el hecho de que sirven de base para los procedimientos de resolución metaheurísticos.

Para realizar esta sección se ha utilizado como base los informes elaborados por el SINTEF[17] sobre la revisión de los dos conjuntos de métodos, y pueden servir de partida para cualquier investigación en esta línea, dada la profundidad con la que se han realizado, como por su actualidad, ya que ambos informes se han presentado a finales del año 2001, por lo que presentan los últimos avances en la materia.

2.10 Formulación del problema

El problema del VRPTW consiste en un conjunto de N clientes, de manera que la denominación de cada uno de los clientes será $1,2,3,...,N$. El conjunto de posibles arcos de conexión entre todos los clientes se denota por A, de manera que no existe ningún arco que comience o termine en el depósito, sino que todos hacen referencia a posibles uniones entre los nodos del problema, excluido el nodo central o depósito. Todas las rutas comienzan y terminan en el depósito central o nodo 0. Existe un coste asociado a cada uno de los arcos en consideración, que se representa por c_{ij} y que hace referencia al coste de viajar del nodo i al nodo j. Este coste está directamente relacionado con el tiempo de viaje entre estos dos nodos, que se representa por el valor t_{ij} y que indica el tiempo de desplazamiento correspondiente al arco $(i,j) \in A$. El tiempo de desplazamiento también incluye el tiempo de servicio del cliente i. El conjunto de vehículos idénticos que componen el problema se denota por V. Cada vehículo tiene una capacidad determinada por q y cada cliente establece una demanda d_i, $i \in N$. Para cada uno de los clientes, el servicio del mismo tiene que comenzar dentro de un intervalo de tiempo denominado ventana de tiempo (*time window*), $[a_i, b_i]$, $i \in N$. Igualmente los vehículos han de salir y llegar al depósito central o nodo 0 dentro de un intervalo de tiempo denotado por $[a_0, b_0]$. En la problemática general del VRPTW está permitido que los vehículos lleguen a los nodos antes de la apertura de la ventana de tiempo, de manera que el tiempo de espera resultante hasta la apertura de la misma no supone ningún coste adicional. Sin embargo no es posible llegar después del cierre de dichas ventanas de tiempo, ni a los clientes en consideración, ni al depósito central.

[17] Véase www.sintef.no/

En este planteamiento general, se establecen dos variables de decisión. En primer lugar, x_{ij}^k definida $\forall (i,j) \in A, \forall k \in V$ que tomaría el valor *1* si el vehículo *k* realiza el desplazamiento entre los nodos *i* hacia el nodo *j*, y tomaría el valor *0,* en caso contrario. La segunda variable de decisión es la variable S_i^k (definida $\forall i \in N, \forall k \in V$) que denota el momento en el que el vehículo *k* comienza el servicio al cliente *i*, de manera que si el vehículo *k* no sirve al cliente *i*, entonces esta variable de decisión no tiene ningún sentido. Igualmente es posible asumir que $S_0^k = 0, \forall k$.

El objetivo del problema es conseguir diseñar un conjunto de rutas que incurran en costes mínimos, una para cada vehículo, de forma que todos los clientes sean atendidos exactamente una sola vez, por lo tanto no se permiten servicios parciales de los mismos. Las rutas han de ser viables con respecto a la capacidad de cada vehículo así como con las ventanas de tiempo de cada cliente. Formalmente el problema se puede describir de la siguiente manera:

$$\min \quad \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}^k$$

Sujeto a

$$\sum_{k \in V} \sum_{j \in N} x_{ij}^k = 1 \quad \forall i \in N$$

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ij}^k \leq q \quad \forall k \in V$$

$$\sum_{j \in N} x_{0j}^k = 1 \quad \forall k \in V$$

$$\sum_{i \in N} x_{ih}^k - \sum_{j \in N} x_{hj}^k = 0 \quad \forall h \in N, \forall k \in V$$

$$\sum_{i \in N} x_{i0}^k = 1 \quad \forall k \in V$$

$$x_{ij}^k (S_i^k + t_{ij} - S_j^k) \leq 0 \quad \forall i, j \in N, \forall k \in V$$

$$a_i \leq S_i^k \leq b_i \quad \forall i \in N, \forall k \in V$$

$$x_{ij}^k = (0,1) \quad i, j \in N, \forall k \in V$$

La función objetivo hace referencia a que todos los costes del problema han de ser minimizados, de manera que los costes los forman únicamente las distancias recorridas

entre los nodos por cada una de las rutas de la solución. La primera restricción establece que todos y cada uno de los clientes han de ser asignados a un vehículo o ruta, es decir que no es posible realizar más de una visita a cada cliente, ni es posible dejar clientes sin visitar, por ello todos los clientes han de estar asignados. La segunda restricción establece que no es posible violar las restricciones de capacidad de los vehículos, de manera que las cargas de los clientes asignados a cada uno de los vehículos no pueden sobrepasar su carga, aunque pueden ser menor que la capacidad del mismo. Las tres siguientes restricciones son las restricciones indicativas del sentido de las visitas en cada una de las rutas de los vehículos, representando que cada vehículo *k* solamente puede abandonar el nodo *0* una vez, abandona el nodo *h* si y sólo si entra en ese nodo, y finalmente regresa de nuevo al nodo nodo *0*. Por último se representan las restricciones relativas a las ventanas de tiempo indicativas de que el vehículo *k* no puede llegar al nodo *j* antes que $S_i^k + t_{ij}$ si viaja de *i* a *j*.

2.11 La evaluación de los métodos heurísticos

La evaluación de los métodos propuestos en la literatura se ha de realizar atendiendo a la comparación de diferentes criterios en la aplicación de los algoritmos. Ejemplos de estos criterios pueden ser el tiempo de ejecución, la calidad de los resultados, la facilidad de implementación, la robustez de los mismos, y por último la flexibilidad a la hora de su aplicación. (Barr et al., 1995; Cordeau et al., 2001). Un punto muy importante en la consideración de la valoración de los métodos heurísticos es su flexibilidad. Si tenemos en cuenta que el desarrollo de los mismos se justifica por su aplicabilidad a problemas del mundo real, entonces es de vital importancia que este tipo de algoritmos sean adaptables a cualquier caso planteado en la realidad. En este sentido ha de ser sencillo el poder establecer cambios en el algoritmo, que nos permitan variar las restricciones del modelo o incluso la función objetivo. Igualmente si nos fijamos en la robustez del algoritmo, es necesario que el algoritmo funcione incluso en circunstancias muy adversas o cuando las restricciones son muy fuertes.

Dos aspectos relevantes en cuanto a la valoración de los métodos de solución son el *tiempo de computación*, y la *calidad* de los resultados aportados por dicho método. Esto se debe a que generalmente existe una relación inversa entre estos dos criterios de valoración que se ha de tener en cuenta. En cuanto a los resultados, se suele medir el grado de cumplimiento de la función objetivo comparando los resultados arrojados por estas técnicas de solución con respecto a la solución óptima, de manera que el algoritmo será tanto mejor cuanto más se aproxime a dicha solución. Sin embargo, la consecución de estas buenas soluciones suele requerir grandes períodos de computación por parte de estos algoritmos, que si bien antaño suponían grandes problemas para los investigadores, recientemente estos problemas se están minimizando debido a los avances en tecnología, concretamente los avances en las velocidades de proceso de los ordenadores.

La medición de la calidad de los resultados aportados por los diferentes métodos se suele realizar por medio de la investigación empírica o experimentación. Generalmente esta fase de la investigación implica probar el algoritmo contra una amplia gama de problemas

tipo recogidos en la literatura, y que sirven de base para la mayor parte de las investigaciones, de manera que se normalizan los resultados para todos los métodos.

A pesar de la existencia de esta normalización en cuanto a los problemas en los que se prueban los algoritmos, todavía existen diferencias derivadas de la tecnología utilizada por cada uno de los autores, y que generalmente implican la utilización de diferentes ordenadores o plataformas de los mismos para la resolución de estos problemas tipo. A pesar de ello se han realizado diferentes estudios con la finalidad de establecer parámetros correctores de las velocidades de cálculo de los diferentes procesadores utilizados en la literatura para establecer comparativas más homogéneas entre los diferentes métodos. En la tabla 2.2 se presentan los parámetros correctores para homogeneizar los resultados de las velocidades de computación indicados en la literatura (Dongarra, 1998).

Procesador	**Mflops/s**
16 x Meiko T-800	7
PC486/66 MHz	1.48
Sun Sparc 10	10
Silicon Graphics 100 MHz	15
Sun Ultra EnterPrise 450	44
8 x Motorola Power PC 604	65
Sun Ultra Sparc 1 167 MHz	70
DEC Alpha	25
HP 9000/720	17
Pentium 200 MHz	24
Pentium 366 MHz	48
Pentium 400 MHz	54
Pentium III 545 MHz	66
Pentium III 800 MHz	100

Tabla 2.2 Factores de ponderación para las velocidades de cálculo (Dongarra, 1998)

Más difíciles de medir son las cuestiones relativas a las técnicas de programación utilizadas por los investigadores, los ajustes realizados en los algoritmos, y los esfuerzos invertidos en su desarrollo (Hooker, 1995). Igualmente existe otro aspecto que complica la comparativa de la valoración de los métodos heurísticos, y que hace referencia a que en la mayoría de los casos los autores solamente informan en sus resultados de las mejores soluciones encontradas durante toda su investigación como medida final de la bondad de sus algoritmos, de manera que no es posible medir los esfuerzos en computación, así como los tiempos invertidos en ajuste en sus algoritmos. Por ello autores como Taillard (2001) defiende la postura de informar también sobre el esfuerzo total en computación tanto a nivel de número de iteraciones necesarias para llegar a los resultados aportados, como a nivel de resultados estadísticos de alcance de dichos resultados.

En el capítulo 4 se incluyen las tablas comparativas entre todos los métodos recogidos en este trabajo para la resolución del VRPTW, así como una comparación entre las velocidades de cálculo de los algoritmos.

2.12 Consideraciones sobre los problemas tipo

Generalmente todos los modelos desarrollados para resolver los problemas de la clase VRPTW se aplican a la batería de problemas estándar desarrollados por Solomon (1987). Por ello en este trabajo también se han utilizado estos problemas tipo para contrastar la eficiencia y eficacia del método presentado.

En este apartado se detalla la composición y los formatos de estos problemas tipo, así como algunas consideraciones a tener en cuenta:

Cada uno de los 56 problemas tipo cuenta con un total de 100 clientes, de manera que el tiempo de desplazamiento entre cada uno de estos clientes es coincidente con la distancia euclídea entre los mismos. Además todos los problemas cuentan con un depósito central del que han de partir los vehículos. Los datos se corresponden con diferentes situaciones de los clientes, así como diferencias temporales entre las ventanas de tiempo y las propias demandas de los clientes.

Cada uno de los problemas contempla siempre una flota de 25 vehículos, lo que nos proporciona un límite superior para la optimización, y una restricción a ser considerada en la programación. Igualmente proporcionan los datos relativos a la capacidad máxima de cada camión.

Los problemas se han dividido en 2 categorías según la dispersión geográfica de los clientes. En primer lugar nos encontramos con los problemas de tipo *R*, que consisten en ejemplos donde los clientes están dispersos aleatoriamente en el plano a través de una distribución uniforme. Por otro lado, existen los problemas de tipo *C*, equivalentes a una disposición de clientes en grupos geográficamente dispersos en el plano. En este caso los clientes están clusterizados en el plano. Por último nos encontramos con los problemas de tipo *RC*, donde se presenta una mezcla de los dos casos anteriores.

Para cada una de las categorías anteriores se presentan dos conjuntos de problemas que se denotan por los subíndices *1* y *2* de manera que en cada uno de los dos subgrupos se presentan entre 8 y 12 problemas individuales.

Los problemas relativos al conjunto *1*, tienen un horizonte de planificación muy breve, de manera que son necesarios múltiples vehículos para realizar los servicios correspondientes a todos los clientes. En este sentido cada ruta considerada debe realizar pocas visitas, y existirá por lo tanto un elevado número de rutas.

Los problemas relativos al conjunto *2* son problemas con horizontes de planificación más amplios, de manera que no son necesarios tantos vehículos como en el caso anterior. Generalmente se trata de problemas cuya solución integra pocas rutas, y éstas son muy largas, incluyendo muchas vistas.

Las diferencias entre estos problemas radican en las demandas de cada uno de los clientes, así como las ventanas de tiempo para los mismos. En cada uno de los subgrupos existen entre 8 y 12 problemas diferentes.

Otra consideración importante es que dentro de todos los subconjuntos la duración de los tiempos de servicio para cualquier cliente, siempre son iguales.

Un estudio más detallado de los problemas tipo desarrollados por Solomon (1987) se recoge en el capítulo 4, relativo a la Experimentación y Resultados del modelo.

En los siguientes apartados se presentan los métodos desarrollados para la resolución del problema del VRPTW, y que comienzan con los algoritmos más sencillos de construcción de rutas, y que mayormente se utilizan en este trabajo como punto de partida para el modelo propuesto.

2.13 Algoritmos de construcción de rutas

Los algoritmos de construcción de rutas trabajan a través de la selección de nodos o arcos de forma secuencial hasta que se consigue una solución factible. Los nodos son escogidos por medio de algún criterio de minimización de coste, sujeto a las restricciones propias de la capacidad del vehículo y de las ventanas de tiempo. Estos métodos pueden trabajar de una manera secuencial, si las rutas se construyen una tras otra, o bien de forma paralela, si las rutas se construyen de forma simultánea. A continuación se presentan los trabajos más destacados en la línea de los algoritmos de construcción de rutas.

Pullen y Webb (1967) describen un sistema desarrollado para la planificación de horarios de trabajo para conductores de furgones en un entorno con grandes restricciones temporales. En este caso, además de las restricciones propias del VRPTW, se consideraron restricciones derivadas de los reglamentos de trabajo de este colectivo. Para ello establecieron un método de construcción paso a paso en el que cada servicio se asignaba al conductor que mejor le cuadraba en su horario, teniendo en cuenta la minimización del coste para aprovechar al máximo los tiempos de los conductores, minimizando los viajes en vacío de los furgones.

Knight y Hofer (1968) presentan un caso de una compañía contratista de servicios de transporte en la que se incluían los servicios de recogida y entrega de forma simultánea. Estos servicios se requerían por teléfono, de manera que se contemplaba la planificación dinámica de los servicios. Para la solución de este caso se plantea un algoritmo que trabaja en dos fases: en primer lugar se asigna la llamada a cada vehículo y, posteriormente, se establecen las órdenes de servicio de las mismas. La asignación de las llamadas a los vehículos se hace en función de un criterio de cercanía, mientras que el orden de servicio de estas llamadas por parte de cada uno de los vehículos se realizaba según la secuencia establecida en la segunda fase. Teniendo en cuenta que solamente había unas pocas opciones en el establecimiento del orden de servicio de las llamadas para cada vehículo, a través de una simple revisión era posible establecer dichos programas de visita. De esta manera se conseguía el objetivo de maximizar el número de

llamadas medias asignadas a cada vehículo así como minimizar el número de vehículos utilizados.

Madsen (1976) desarrolla dos algoritmos para la resolución del VRPTW con ventanas muy rígidas para ser aplicados al sector de distribución de prensa y revistas. El enfoque adoptado no contempla las limitaciones de capacidad de la flota de vehículos, ni las ventanas de tiempo más tempranas. Este autor propone dos métodos: el primero hace referencia a la inserción en paralelo de nodos según sus momentos de cierre y las distancias de los clientes a los vehículos y el segundo se fundamenta en la simulación Monte Carlo para la resolución del problema a través de la asignación de los pedidos arbitrariamente a las rutas, ordenando los clientes en cada ruta de acuerdo a sus momentos de cierre. Finalmente la solución se mejoraba con reinserciones simples entre rutas de los clientes no atendidos.

Solomon (1986) establece para la resolución del VRPTW un primer método consistente en la creación de una única ruta en primer lugar, y después su división en subrutas para respetar las ventanas de tiempo. Este autor propone la creación de un primer programa de servicio con todos los clientes ordenados por sus ventanas de tiempo, para posteriormente dividir esta ruta en otras más pequeñas, aplicando los criterios de capacidad de los vehículos. Sin embargo en este caso no aporta resultados computacionales.

Sin embargo en Solomon (1987) el autor propone tres nuevos métodos para la resolución del VRPTW, y que servirán de base para la gran mayoría de los trabajos en esta línea de investigación.

El primero de estos métodos consiste en una extensión del conocido método de los ahorros de Clarke y Wright (1964). Este método fue desarrollado inicialmente para la resolución del VRP, y constituye uno de los más conocidos métodos de construcción de rutas. Comienza con una solución en la que cada cliente constituye una única ruta servida por un único vehículo individual. A partir de esta solución inicial los clientes se integran en rutas más grandes a través del cálculo de los ahorros generados por estas posibles combinaciones y que equivalen a $S_{ij} = d_{i0} + d_{0j} - d_{ij}$. De esta manera Clarke y Wright seleccionan los arcos *(i,j)* que provocan los máximos ahorros para unir clientes en rutas más grandes. A través de sucesivas iteraciones es posible ir construyendo las rutas finales, eliminando las rutas iniciales de clientes únicos, siempre que las soluciones aportadas sean respetuosas con las restricciones del problema. A la hora de combinar estas rutas en rutas más grandes es posible aplicar dos criterios diferentes: o bien realizar combinaciones paralelas en todas las rutas generadas al mismo tiempo, o bien tratar de completar una ruta inicialmente escogida hasta que el vehículo esté completamente lleno. Es decir, que se podrían implementar tanto algoritmos de tipo secuencial, como algoritmos de tipo paralelo. Igualmente Solomon contempla en su algoritmo la cercanía temporal y la cercanía geográfica, algo que no contemplaban Clarke y Wright en su algoritmo, dado que inicialmente estaba destinado a la resolución del VRP sin ventanas de tiempo. Para ello, y ante la consideración de las diferentes uniones y los ahorros consiguientes, el algoritmo realiza la comprobación de las restricciones de temporalidad

para eliminar aquellas uniones que no son posibles. De esta manera se cumplen todas las restricciones impuestas tanto por las ventanas de tiempo de los nodos, como de la temporalidad del horizonte de planificación.

El segundo método que propone Solomon consiste en un método de adición del cliente más cercano a la ruta, pero en este caso incluyendo el criterio temporal. Los pasos que sigue el algoritmo son muy intuitivos. Cada ruta comienza con la selección del cliente que esté más cercano al depósito central. Posteriormente en cada iteración del algoritmo se seleccionan los clientes más cercanos a los últimos clientes añadidos a las rutas, añadiendo éstos a las mismas. Una nueva ruta se genera cuando no es posible incluir a un cliente seleccionado en ninguna posición factible. Al igual que en el caso anterior, el concepto de cercanía incluye tanto el criterio espacial como el criterio temporal.

Por último, el tercer y más exitoso método heurístico propuesto por Solomon es el denominado *I1*. Los pasos del mismo comienzan con la selección de un cliente semilla a partir del cual se añaden el resto de los clientes de la ruta hasta que ésta agota su horizonte temporal, o bien se cubre la capacidad del vehículo. Si quedan clientes sin asignar, entonces se repiten los pasos descritos, hasta que se agotan los clientes. Los clientes semilla seleccionados son aquellos que, o bien son los más distantes del depósito central, o bien los que proporcionan un momento de apertura más temprano para realizar el servicio.

Después de iniciar una ruta con uno de estos clientes semilla, el algoritmo aplica dos criterios de inserción, $c_1(i,u,j)$ y $c_2(i,u,j)$ para seleccionar la inserción del cliente u entre los clientes adyacentes i y j. Sea $(i_0,i_1,i_2,\ldots,i_m)$ la ruta donde i_0 e i_m representan al depósito. Para cada cliente no asignado u, se calcula el mejor coste de inserción en las rutas de la siguiente manera:

$$c_1(i(u),u,j(u)) = \underset{\rho=1,\ldots,m}{opt}\ c_1(i_{\rho-1},u,i_\rho)$$

A continuación, el mejor cliente u^* para ser insertado en una ruta es aquél para el que

$$c_2(i(u^*),u^*,j(u^*)) = \underset{u}{opt}\, c_2(i(u),u,j(u))$$

donde u es un cliente no asignado a ninguna ruta, y además su posible inserción en la ruta es factible con las restricciones del problema.

Entonces el cliente u^* se inserta en la ruta entre los clientes $i(u^*)$ y $j(u^*)$. Cuando no hay más clientes para ser insertados en las rutas existentes, entonces el algoritmo genera una nueva ruta, a no ser que ya se hayan insertado todos.
Para el cálculo de $c_1(i,u,j)$ se tiene en cuenta lo siguiente:

$$c_1(i,u,j) = \alpha_1 c_{11}(i,u,j) + \alpha_2 c_{12}(i,u,j)$$

en donde

$$\alpha_1 + \alpha_2 = 1, \quad \alpha_1 \geq 0, \quad \alpha_2 \geq 0$$

$$c_{11}(i,u,j) = d_{iu} + d_{uj} - \mu d_{ij}, \quad \mu \geq 0$$

$$c_{12}(i,u,j) = b_{ju} - b_j$$

siendo d_{iu}, d_{uj}, y d_{ij} son las distancias entre los clientes *i* y *u*, *u* y *j*, e *i* y *j* respectivamente. El parámetro μ controla la importancia que tienen los ahorros en la distancia y b_{ju} denota el nuevo momento de llegada al cliente *j*, suponiendo que el cliente *u* se haya insertado en la ruta mientras que b_j era el momento de llegada antes de la inserción.

De forma conceptual, el criterio de decisión c_{11} hace referencia al valor del incremento en la distancia original provocados por la inserción del nodo *u,* entre los nodos *i* y *j.* Además Solomon establece un parámetro de valoración de la importancia de este incremento de distancia en la decisión. Es decir que establece una penalización más o menos elevada a estos desplazamientos adicionales. Igualmente el criterio de c_{12} hace referencia al retraso temporal en la llegada al cliente *j* provocado por la visita intermedia al cliente *u*. En este sentido Solomon contempla como criterio de decisión el que estas visitas intermedias no repercutan excesivamente en los retrasos a los nodos posteriores.

De forma análoga, al incluir estos dos factores de decisión en un único criterio c_1, es posible parametrizarlos de forma que se otorga mayor o menor importancia a cada uno de ellos a través de los factores de ponderación.

El criterio de inserción c_2 se calcula de la siguiente manera:

$$c_2(i,u,j) = \lambda d_{ou} - c_1(i,u,j), \quad \lambda \geq 0$$

donde el parámetro λ se utiliza para ponderar la importancia de mantener el nodo como una ruta separada, frente a la opción de realizar definitivamente la inserción.

Solomon (1987) propone otros dos métodos de inserción denominados *I2*, en el que las inserciones de clientes se realizan en función a la minimización de la distancia y tiempo total de las rutas, e *I3*, que tiene en cuenta la urgencia de servicio de los clientes.

Dullaert (2000a y 2000b) establece que el criterio de inserción de Solomon $c_{12}(i,u,j)$ subestima el tiempo adicional que supone la inserción del cliente *u* entre el depósito y el primer cliente en las rutas en construcción. Esto puede provocar que se seleccionen subóptimos en las inserciones de estos clientes, por lo que una ruta con un número reducido de clientes puede tener un programa de servicios muy disperso en el tiempo. Este autor introduce un nuevo criterio de inserción que contempla la variable temporal de manera que proporciona unos resultados en cuanto a los ahorros generados de

aproximadamente el 50%. Sin embargo estos ahorros decrecen a medida que se aumenta el número de clientes de las rutas.

Solomon (1987) también propone una variante del conocido algoritmo de barrido de Gillett y Miller (1974) a través de la descomposición del problema general del VRPTW en dos fases bien diferenciadas. En la primera fase los clientes se asignan a las rutas, tal y como se hace en el algoritmo original de barrido. En este caso el centro de gravedad es calculado de manera que los clientes se agrupan de acuerdo con su ángulo polar. En la segunda fase los clientes son asignados a un vehículo utilizando el criterio de inserción del tipo *I1*.

Potvin y Rousseau (1993) presentan una versión paralelizada del método de inserción de Solomon *I1*, donde un conjunto de *m* rutas se inician simultáneamente. Los autores utilizan el heurístico de inserción de Solomon para determinar el número inicial de rutas y el conjunto inicial de clientes semilla. La selección del cliente siguiente a ser considerado se basa en una medida del rechazo de inserción en todas las rutas, de manera que un gran rechazo se produce cuando existe un gran espacio entre el mejor y el segundo mejor puesto para ser insertado un cliente.

Foisy y Potvin (1993) implementan la versión paralelizada descrita en Potvin y Rousseau (1993) en un cluster de 2-6 estaciones de trabajo *Sun 3*. El paralelismo se implementa en el cálculo de los costes de inserción para cada cliente. En este sentido se presenta un trabajo meramente de distribución de cargas de computación, concluyendo los autores que la reducción total en tiempo de computación es lineal con el número de procesadores para la parte distribuida del algoritmo heurístico.

Ioannou et al. (2001) utilizan el criterio de inserción secuencial propuesto por Solomon (1987) para resolver los problemas de tipo teórico propuestos en la literatura, así como un ejemplo práctico de una empresa de transporte de comida. El enfoque presentado por estos autores recoge la idea de que la selección de los clientes a ser insertados, debe de estar motivada por la función que minimiza el impacto sobre los clientes que ya están incluidos en la ruta en construcción, así como el impacto sobre los clientes todavía no incluidos, y el impacto sobre las ventanas de tiempo del cliente a ser insertado.

Balakrishnan (1993) describe tres métodos heurísticos para la resolución de VRPs con ventanas de tiempo poco rígidas (*VRP with Soft Time Windows*, o VRPSTW). Los heurísticos recogidos en este trabajo se fundamentan en el algoritmo de cliente más cercano (*Nearest Neighbor*) y en las reglas del algoritmo de Clarke y Wright, pero se diferencia de estos en la forma de seleccionar el primer cliente de una ruta, así como en los criterios utilizados para escoger el siguiente cliente para ser insertado. La motivación del VRPSTW se fundamenta en que, si es posible obviar algunas de las restricciones de las ventanas de tiempo propuestas en el modelo original, es posible obtener importantes ahorros en el cómputo global de la solución, tanto en la distancia total recorrida, como en el número de vehículos utilizados. En este sentido la aplicabilidad de este tipo de problemas a la realidad empresarial cobra una gran importancia dado que en ocasiones la

existencia de ventanas de tiempo no supone restricciones inviolables por parte de los servicios de las empresas, sino que constituyen simples aproximaciones que son útiles para el planificador. Entre las aplicaciones más comunes del VRPSTW está el problema conocido por "*dial a ride*", consistente en la planificación dinámica de servicios realizados por una flota de vehículos y que se fundamenta en el hecho de que las órdenes de servicio van apareciendo a medida que se están llevando a cabo otros servicios.

Bramel y Simchi-Levi (1996) proponen un método heurístico asintóticamente óptimo que se basa en la idea de resolver el problema de concentración de cargas con ventanas de tiempo (*Capacited Location Problem with Time Windows*). En este problema el objetivo consiste en seleccionar un conjunto de posibles lugares en los que posicionar o enviar un vehículo para que realice los servicios al resto de los clientes desde esa posición. En el contexto del VRPTW la selección de los lugares en los que posicionar un vehículo se correspondería a la selección de un conjunto de clientes semilla desde los que se iniciarían las rutas. Los autores utilizan la relajación de la lagrangiana para resolver el CLPTW, mientras que el resto de los clientes son insertados a través de un procedimiento de tipo "*greedy*" de manera que sus inserciones minimicen las distancias totales recorridas por cada ruta. Los autores concluyen que su heurístico mejora 25 del los 56 soluciones de los problemas tipo de Solomon encontrados hasta la fecha, utilizando tiempos razonables de computación.

2.14 Procedimientos de mejora de rutas.

Russel (1977) analiza los primeros trabajos sobre el VRPTW para los heurísticos de mejora a través de los intercambios de arcos para conseguir soluciones *k*-óptimas. El enfoque así denominado de *M*-Rutas era muy eficaz resolviendo problemas con pocos clientes integrados en ventanas de tiempo. Por ejemplo, la solución *k*-óptima para un problema con 163 clientes, donde solamente el 15% de los clientes estaban condicionados por ventanas de tiempo, tardaba unos 90 segundos en un IBM 370/168.

Posteriormente se realizaron aplicaciones más eficientes acelerando la revisión de soluciones no factibles y la evaluación de la propia función objetivo, tal y como se recoge en Savelsbergh (1986), Solomon y Desrosiers (1988), Solomon, Baker y Schaffer (1988), Savelsbergh (1990) y Savelsbergh (1992). Las técnicas más utilizadas se basaban en preproceso de datos para el problema, mecanismos de actualización y estrategias de búsqueda lexicográfica.

Baker y Schaffer (1986) presentan un estudio empírico sobre los procedimientos de mejora de rutas que pueden ser aplicados a los procedimientos heurísticos de construcción. El algoritmo del *Nearest Neighbor* con ventanas de tiempo, así como tres heurísticos de Inserción más Económica se presentan en este trabajo como algoritmos de construcción de rutas. Los algoritmos de inserción trabajan con diferentes funciones de coste: la distancia, el incremento en el tiempo de llegada, el tiempo de espera, al igual que en los trabajos originales de Solomon (1987). Los métodos de mejora considerados son los mecanismos de intercambio 2-opt y 3-opt diseñados originalmente por Lin (1966), pero adaptados para el caso de las ventanas de tiempo. Se prueban tanto los intercambios

intra-ruta, como inter-ruta. Una conclusión interesante de los autores es que las soluciones mejores conseguidas son aquellas que parten de las mejores soluciones iniciales, y que se fundamentan normalmente en los procedimientos de Inserción más Económica, mejorando los del Vecino más Cercano.

Landeghem (1988) presenta un método heurístico con dos criterios fundamentado en el algoritmo de los ahorros de Clarke y Wright (1964). El autor propone la combinación de las medidas originales de los ahorros generados en términos del tiempo ahorrado, con otra medida de "pérdida de flexibilidad". La flexibilidad la define como la diferencia entre la ventana de tiempo del cliente y la ventana de tiempo de la ruta después de la inserción. La ventana de tiempo la ruta se refiere a la diferencia entre los períodos de tiempo en el que un vehículo puede comenzar el servicio del primer y del último cliente en una ruta. Al final, los resultados son mejorados utilizando reinserciones simples de clientes.

Potvin y Rousseau (1995) comparan diferentes heurísticos de intercambio de arcos para el VRPTW (2-opt, 3-opt y Or-opt) e introducen un nuevo mecanismo 2-opt*. La base para este nuevo mecanismo de intercambio es la combinación de dos rutas de forma que los últimos clientes de una ruta se introducen después de los primeros clientes de otra ruta, permaneciendo el sentido de las mismas. Este mecanismo se ilustra en la figura 2.17.

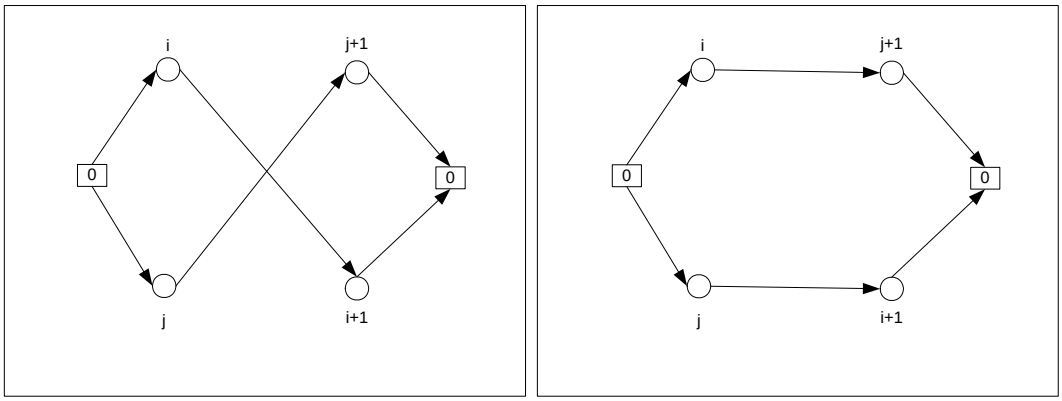


Figura 2.17 El operador 2-opt*. Los nodos servidos después del cliente *i* en la ruta superior se reinsertan para ser servidos por la ruta inferior después del cliente *j* de forma que los clientes que antes eran servidos después de *j* se mueven a la ruta superior para ser servidos después de *i*. Esto se consigue a través del cambio de los arcos *(i, i+1)* y *(j, j+1)* por los arcos *(i, j+2)* y *(j, i+1)*

Como caso especial de este mecanismo se destaca la posibilidad de que dos rutas se combinen en una sóla si el arco *(i,i+1)* es el primero de su ruta, y el arco *(j,j+1)* es el último en la suya, o viceversa. También se destaca que el enfoque combinado de los mecanismos 2-opt* y Or-opt resulta muy potente. Otro dato a destacar es que las soluciones iniciales se producen siempre con el heurístico de Solomon *I1*.

Prosser y Shaw (1996) también establecen una comparación del mecanismo intra-ruta 2-opt de Lin (1966) con los mecanismos inter-ruta de Reposicionamiento, de Intercambio y los mecanismos de Cruce, originalmente propuestos por Savelsbergh (1992) para el

problema del VRP, y que se recogían en la sección anterior. El mecanismo de 2-opt funciona a través de la reversión de parte de una ruta sencilla. El mecanismo de Reposicionamiento simplemente desplaza un cliente de su posición en una ruta a otra ruta. El mecanismo de Intercambio realiza permutas de dos visitas en diferentes rutas. Por último el mecanismo de Cruce es similar al mecanismo 2-opt* propuesto por Potvin y Rousseau (1995) para el VRPTW. Inicialmente existe un vehículo virtual que realiza todas aquellas visitas que no se han realizado por los vehículos reales. Este vehículo virtual tiene dos características que lo definen. La primera obedece a que puede realizar un número infinito de visitas a clientes, por lo que no tiene restricciones de capacidad, y en segundo lugar, que el coste de las visitas de este vehículo a los clientes es artificialmente más elevado que el de los vehículos reales.

Thompson y Psaraftis (1993) desarrollan un método fundamentado en *k*-transferencias cíclicas que implican las transferencias simultáneas de *k* demandas de la ruta I^j a la ruta $I^{\delta(j)}$ para cada *j* y cada integral fijo *k*. El conjunto de rutas $\{I^r\}$, $r=1,...,m$ constituyen una solución factible, y δ es una permuta cíclica de un suconjunto de $\{1,...,m\}$. En concreto, cuando δ tiene una cardinalidad fija *C*, es posible obtener una *k*-transferencia *C*-cíclica. Si se permiten *k* demandas ficticias en cada ruta, las transferencias de demandas se pueden obtener a través de permutas más que de permutas cíclicas de rutas. Debido a la complejidad del mecanismo de transferencias cíclicas, éste se desarrolla de forma heurística. Para una mayor comprensión este mecanismo se presenta la figura 2.18, más concretamente para el caso del operador de 2-transferencias 3-cíclicas.

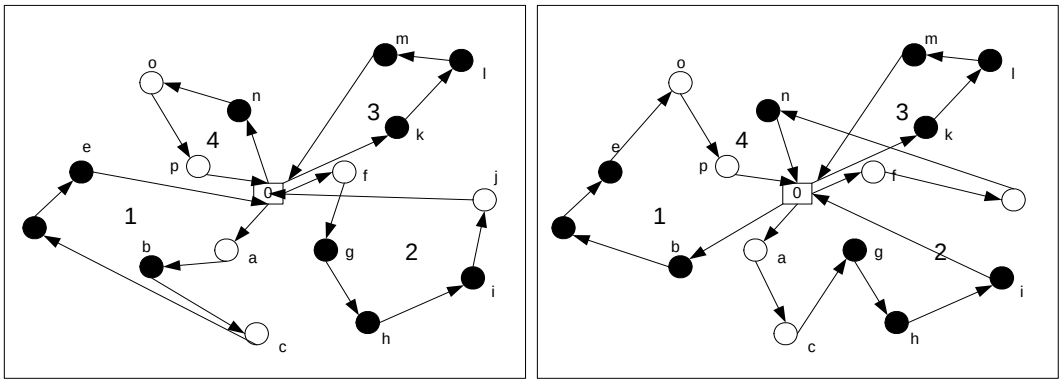


Figura 2.18 El operador de transferencia cíclica. La idea principal consiste en transferir simultáneamente los clientes representados en blanco de una forma cíclica entre las rutas. En este ejemplo, los clientes *a* y *c* de la ruta *1*, *f* y *j* de la ruta *2*, y *o* y *p* de la ruta *4* son transferidos simultáneamente a las rutas *2*, *4* y *1* respectivamente, mientras que la ruta *3* permanece inalterada.

Antes y Derigs (1995) proponen un enfoque de construcción paralela que construye y mejora múltiples rutas simultáneamente. El enfoque se basa en el concepto de negociación entre clientes y rutas. Primero, cada uno de los clientes no incluidos en rutas solicitan un coste de servicio a cada una de ellas, y segundo, cada ruta selecciona la proposición más eficiente. Los precios se calculan de acuerdo con las medidas de Solomon para la inserción (heurístico *I1*). Una vez construida una solución factible, el número de rutas se reduce en una, y el problema se vuelve a resolver. Los autores

también proponen un algoritmo de postoptimización, donde algunos de los clientes más ineficientes se retiran de las rutas, para posteriormente reinsertarlos a través del procedimiento de negociación descrito anteriormente.

Russell (1995) combina los procedimientos de mejora de rutas, con los de construcción propiamente dichos. El procedimiento de construcción es similar al propuesto por Potvin y Rousseau (1993). Primero se seleccionan N puntos semilla representando clientes ficticios a través del procedimiento de generación de puntos semilla de Fisher y Jaikumar (1981), propuesto originariamente para el VRP. La idea consiste en generar los puntos semilla o de partida a través del estudio de la capacidad de los vehículos para así crear sectores y decidir las distancias de las semillas a los depósitos para cada sector. Para la selección del siguiente cliente a ser insertado se presentan tres mecanismos de pedido, que son: la ventana más temprana, la distancia más lejana desde el depósito, la amplitud de la ventana aumentada en la distancia desde el depósito. El método de búsqueda local se fundamenta en el esquema presentado por Christofides y Beasley (1984), donde un movimiento se genera a través de la eliminación y posterior reinserción de 4 clientes cercanos los unos a los otros. Para cada cliente se determinan primero las dos mejores rutas según el coste de inserción propuesto por Solomon. Esto solamente se realiza para las dos mejores rutas, dado que el cómputo para todas ellas sería muy costoso en términos de tiempo de computación. Este intercambio se realiza después de que se hayan incluido en las rutas un número k de clientes. Este enfoque es comparable al heurístico k-opt de intercambio de ramas de múltiples rutas de Russell (1977). El autor establece finalmente que la combinación simultánea de los métodos de construcción de rutas y de mejora de las mismas es una opción más eficiente que los métodos tradicionales de construcción primero y mejora después.

Thangiah et al. (1995) estudia los problemas de VRP con plazos de entrega (VRP *with Time Deadlines*, VRPTD), es decir, únicamente con momentos de cierre en las ventanas de tiempo. En este sentido los autores desarrollan dos métodos heurísticos fundamentados en los principios del heurístico de Barrido orientado por tiempo, y de los mecanismos de inserción más económica para la resolución del VRPTD, cumplimentados por el algoritmo de λ- intercambios de Osman (1993). Los autores concluyen que ambos heurísticos trabajan bien en problemas donde están agrupados, o bien en aquellos donde los plazos de entrega son muy amplios, algo también bastante intuitivo.

Hamacher y Moll (1996) describen un heurístico para la aplicación a VRPs en la vida real con ventanas de tiempo muy estrechas, para los envíos de verdura a los restaurantes. El algoritmo se divide en dos partes. En primer lugar se trata de agrupar los clientes en subconjuntos utilizando un criterio puramente geográfico a través de la estructura del mínimo árbol de cobertura (*Minimum Spanning Tree, MST*). El MST se subdivide en pequeñas ramas donde los nodos de cada una de las ramas representa a los clientes pertenecientes a una ruta. A continuación se contempla la creación de estas rutas atendiendo a múltiples criterios como el número de clientes, distancia, demanda total, y los tipos de ventanas de tiempo. Finalmente los clientes de estos clusters se incluyen en las rutas a través de la programación de sus visitas según el algoritmo de Inserción más

Económica seguido de mejoras locales a través de la eliminación de pequeños fragmentos de la ruta, y posteriores reposicionamientos de los mismos. Por último si no es posible encontrar una ruta factible, los clientes restantes se asignan manualmente.

Shaw (1997) realiza sus estudios en la búsqueda de amplios vecindarios (*Large Neighborhood Search, LNS*), a través de la reprogramación de clientes seleccionados a través de las técnicas de programación de restricciones. LNS supone una problemática similar a la de programación de tareas en plantas de producción (Applegate y Cook, 1991), que a su vez se fundamenta en los problemas de gestión de los cuellos de botella en procesos de fabricación y montaje, como puede observarse en Adams et al. (1988). El procedimiento descrito por Shaw consiste en la selección aleatoria de un número determinado de clientes para ser retirados de sus rutas originales, de manera que después se reinsertan de forma óptima en las rutas preexistentes. Para que estos intercambios sean factibles la selección de los clientes se realiza de forma que sean intercambiables entre sí, bien porque están geográficamente cercanos, bien porque están servidos por el mismo vehículo, bien porque tienen demandas similares, y además porque tienen ventanas de tiempo similares en cuanto a su apertura. A continuación se aplica un método de acotación combinado con la programación de restricciones para reprogramar los clientes previamente retirados. En la solución original este autor propone que cada cliente es visitado por un solo vehículo. Debido a los requerimientos computacionales tan elevados, este tipo de enfoques solamente es aplicable a problemas con un reducido número de nodos en sus rutas.

Cordone y Wolfer-Calvo (1998) proponen un método heurístico determinista basado en los intercambios del tipo *k*-opt combinado con un procedimiento para reducir el número de rutas. La característica especial del algoritmo es que alterna entre la minimización de la distancia total y la minimización de la duración total de la ruta para escapar de los óptimos locales. El algoritmo construye un conjunto inicial de rutas a través del algoritmo de inserción de Solomon *I1*, aplica un procedimiento de búsqueda local (intercambios de 2 ó 3 arcos) y escoge el mejor. El procedimiento de reducción de rutas intenta insertar a cada uno de los clientes de una ruta al mismo tiempo que en otras. Si este procedimiento falla, entonces se intenta un procedimiento de segregación de cadenas, donde un cliente c_j se retira primero de una ruta r_n y se inserta en una nueva ruta r_m, antes de insertar el cliente c_i en la ruta r_n. Los autores también plantean una serie de técnicas especiales para reducir los tiempos de computación. La primera técnica consiste en la creación de macronodos, consistentes en agrupaciones de nodos adyacentes con la finalidad de considerarlos como un único nodo para facilitar los cálculos. Otras técnicas consisten en explorar el vecindario de *k* nodos en un orden lexicográfico y mantener en memoria al mejor intercambio de cada ruta, de cada par y de cada trío de rutas.

Caseau y Laburthe (1999) proponen un modelo especialmente diseñado para problemas de planificación de rutas muy grandes. Los autores introducen una variante al mecanismo de Inserción más Económica que alterna entre la mejor y la segunda mejor ruta para cada uno de los clientes, siempre que la diferencia entre ambas sea pequeña. Durante la construcción de la solución, se consideran tres tipos de movimientos para cada una de las

inserciones: el mecanismo de intercambio 2-opt*, la Reinserción de una cadena de clientes consecutivos de una ruta *r* a una ruta *r′*, así como el mecanismo de Transferencia simple de un cliente. Cuando no se encuentra un lugar para insertar a ese cliente, entonces se consideran tres tipos de movimientos para crear esa posición de inserción. El primer mecanismo es de Permuta, y consiste en retirar una cadena de clientes consecutivos de una ruta *r* para añadirlo a una ruta *r′.* El segundo de estos mecanismos consiste en el reposicionamiento de un nodo retirándolo de un vértice de la ruta *r,* y reposicionándolo en otro vértice de una ruta *r′.* El último de estos mecanismos consiste en el vaciado de una ruta y el reposicionado de todos los nodos de la misma en otras rutas, siempre que esto sea posible. En aquellos casos en los que el número de clientes de una ruta sea menor que 30, el orden de los clientes de la misma es optimizado utilizando el algoritmo exacto de *Branch and Bound* basado en restricciones propuesto por Caseau y Laburthe (1997). Caso contrario, para rutas más largas, se utiliza un mecanismo de 3-opt para modificar las rutas después de cada inserción. Los autores también procuran restringir la inserción de los clientes a una ruta empleando criterios geográficos.

Hong y Park (1999) proponen un algoritmo de dos fases consistente en un método de inserción paralela para agrupar los nodos, así como un procedimiento de programación lineal de objetivos para generar las rutas. El primer criterio del algoritmo busca la minimización de la distancia total recorrida en vez del número de vehículos utilizados, mientras que el segundo criterio procura la minimización del tiempo total de espera ante los nodos. Los clientes semilla se seleccionan a través de la identificación de aquellos que no pueden ser servidos por la misma ruta debido a las restricciones temporales. El resto de los clientes se insertan a estas rutas inicializadas de manera que el incremento de la distancia total de la ruta y el tiempo de espera es mínimo. Al igual que ocurría en Potvin y Rousseau (1993), los clientes con pocas posibilidades de inserción y grandes diferencias entre el mejor y el segundo mejor puesto para insertar son considerados en primer lugar. Al final de la fase de agrupación, se reforman los grupos a través de los procedimientos de intercambio de 2-opt y Or-opt. En la fase de generación de las rutas, el modelo de programación de objetivos se descompone en dos modelos de programación lineal, donde se procura minimizar o bien la distancia total recorrida, o bien el tiempo total de espera. Estos autores consiguen resultados algo mejores que Potvin y Rousseau, pero utilizando tiempos superiores de computación.

Bräysis (2001a) describe numerosos heurísticos de búsqueda local utilizando un nuevo enfoque de tres fases para el VRPTW. En la primera fase, se generan un elevado número de soluciones iniciales utilizando heurísticos de construcción de rutas a través de la combinación de diferentes parámetros. En la segunda fase se intenta reducir el número de rutas utilizando un mecanismo de Eyección de Cadenas de nodos (Glover, 1991 y 1992) que también procura reordenar las rutas. En la tercera y última fase, se utiliza el mecanismo de intercambio Or-opt para minimizar la distancia total recorrida. Uno de los mecanismos de construcción de rutas se fundamenta en los principios de Solomon (1987) y Russell (1995), donde las rutas se generan de una en una de manera secuencial y después de que un número *k* de nodos hayan sido insertados en la ruta, se procede a la reordenación de los mismos a través del mecanismo Or-opt. Además se introducen

nuevas reglas de selección de clientes semilla. El otro mecanismo heurístico de construcción se basa en el método de los ahorros de Clarke y Wright (1964). En este trabajo se implementa una versión paralelizada del método de los ahorros, introduciendo una nueva medida para el cálculo de los mismos para contemplar a su vez los tiempos de espera resultantes.

Bräysis (2001b) sugiere posteriormente dos nuevos heurísticos especialmente diseñados para los problemas clusterizados de VRPTW. El primer enfoque es similar al caso presentado en las líneas anteriores, aunque la diferencia básica es la utilización de cuatro nuevos mecanismos de búsqueda local en vez del mecanismo de intercambio Or-opt en la tercera fase de construcción. Estos mecanismos se basan en los trabajos de Taillard con su mecanismo de *CROSS-intercambios*. El segundo método consiste en identificar clientes dentro del mismo cluster a través de la generación de zonas o cajas alrededor de los clientes semilla identificados. Los clientes dentro de estas zonas se reordenan utilizando los mecanismos de Inserción más Económica y los mecanismos de intercambio de Or-opt, según los valores de sus ventanas de tiempo. A mayores, si algún cliente no se ubica en ninguna de las zonas de inserción, entonces se intenta asignarlo a alguna ruta a través del procedimiento de Inserción más Económica. Al final también se aplican mecanismos de intercambio entre rutas. Los resultados de este mecanismo propuesto por el autor se muestran enormemente eficientes, consiguiendo obtener la solución óptima para los 17 problemas de tipo Clusterizado de Solomon (1987), en solamente 1 segundo de computación.

2.15 Procedimientos compuestos: metaheurísticos

En las secciones anteriores para el TSP y el VRP se presentaban los algoritmos metaheurísticos como procedimientos para exploran el amplio espacio de soluciones posibles para identificar aquellas soluciones que eran buenas y que generalmente parten de los mecanismos presentados anteriormente de construcción de rutas así como los algoritmos de mejora. La característica fundamental es que permiten el empeoramiento de las soluciones de partida, así como la violación de las restricciones con la única finalidad de huir de los óptimos locales para seguir buscando el óptimo global. En este apartado se describe la aplicación de estas técnicas al caso del VRPTW.

2.15.1 Búsqueda tabú.

García et al. (1994) son los primeros en implementar las técnicas de búsqueda tabú para la resolución del VRPTW. Los autores presentan una implementación paralelizada en una red de 16 *transputers T-800 de Meiko*. La búsqueda tabú que implementan es bastante sencilla, ya que se fundamenta en la aplicación del método heurístico de Solomon *I1*, para generar una solución inicial, y a partir de ahí aplican los procedimientos de mejora de 2-opt y Or-opt.

A partir de ese momento muchos autores han presentado diferentes implementaciones de estas técnicas a través de la utilización de procedimientos de intensificación y diversificación más sofisticados, estrategias para minimizar el número de rutas, técnicas

complejas de post-optimización, hibridizaciones con otras técnicas de búsqueda como el recocido simulado o los algoritmos genéticos, implementaciones paralelizadas y la permisión de violación de restricciones durante las búsquedas.

La solución inicial generalmente se obtiene a través de algún algoritmo de inserción sencillo, como el de inserción más económica descrito anteriormente. El más común es el algoritmo *I1* de Solomon, descrito en el apartado anterior. Sin embargo algunos autores proponen métodos alternativos para la generación de estas soluciones iniciales como Chiang y Russell (1997), donde se aplica una versión paralelizada del algoritmo de Russell (1995). De Backer et al. (1997) y Schulze y Fahle (1999) utilizan el algoritmo de los ahorros de Clarke y Wright (1964); Tan et al. (2000) utilizan una versión del algoritmo de Solomon modificada propuesta por Thangiah (1994) y Cordeau et al. (2001) utilizan una versión modificada del algoritmo de Gillet y Miller de barrido (1974).

Después de crear una solución inicial, se intenta mejorar a través de procedimientos de búsqueda local en el vecindario, así como los criterios de aceptación de la mejor opción. La mayor parte de estos vecindarios de búsqueda se construyen con procedimientos de mejora descritos en el punto anterior: 2-opt, Or-opt, 2-opt*, Reposicionamiento, Intercambio, etc. Otros procedimientos son el λ-intercambio de Osman (1993), el intercambio *CROSS* de Taillard et al. (1997), el intercambio *GENI* de Gendrau et al. (1992) y las eyecciones de cadenas de Glover (1991 y 1992).

El mecanismo de generación λ-intercambio se puede describir de la siguiente manera: dada una solución para el problema representada por un conjunto de rutas $S=\{r_1, ..., r_p, ..., r_q, ..., r_k\}$, un λ-intercambio consiste en una permuta para un par de rutas (r_p, r_q) de un conjunto de clientes S_1 pertenecientes a la ruta r_p de tamaño menor o igual que λ, por otro conjunto de clientes S_2 pertenecientes a la ruta r_q de tamaño menor o igual que λ, de manera que se generan dos nuevas rutas $r_p^* = (r_p - S_1) \cup S_2$, $r_q^* = (r_q - S_2) \cup S_1$ generando una nueva solución $S=\{r_1, ..., r^*_p, ..., r^*_q, ..., r_k\}$. El vecindario $N_\lambda(S)$ de una solución determinada S es el conjunto de todas las soluciones vecinas S' generados para un valor concreto λ.

En el intercambio de tipo *CROSS*, la idea es mover dos arcos $(i-1, i)$ y $(k, k+1)$ de una primera ruta, mientras dos arcos $(j-1, j)$ y $(l, l+1)$ son retirados de una segunda ruta. A continuación los segmentos i-k y j-l, que pueden contener arbitrariamente un número determinado de clientes, son intercambiados a través de la introducción de los nuevos arcos $(i-1,j), (l, k+1), (j-1, i)$ y $(k, l+1)$, tal y como se ilustra en la figura 2.19.

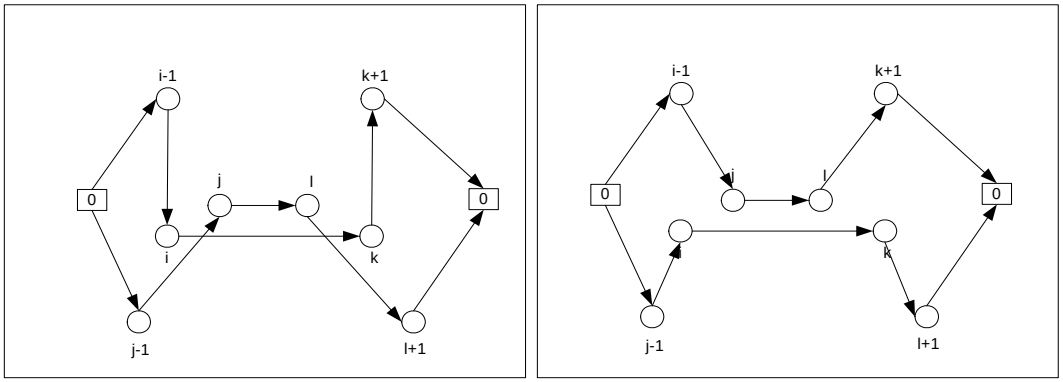


Figura 2.19 Operador de intercambio tipo *CROSS*. Los segmentos *(i, k)* de la ruta superior y *(j ,l)* de la ruta inferior se reinsertan simultáneamente en las rutas inferior y superior respectivamente. Esto se consigue a través del intercambio de los arcos *(i-1, i), (k, k+1) (j-1, j)* y *(l, l+1)* por los arcos *(i-1, j), (l, k+1), (j-1, i)* y *(k, l+1).*

La idea de la eyección o segregación de cadenas de nodos consiste en retirar algún cliente c_i de una ruta r_k y después insertar otro cliente c_j actualmente servido por una ruta r_l en la ruta parcial r_k. Después se intenta insertar el cliente c_i en otra ruta r_m, de forma que $r_m \neq r_k$. Si se encuentra una mejoría, entonces se completa la cadena y otro cliente c_{j+1} es seleccionado para iniciar otra cadena. En cada fase de la eyección de cadenas, un cliente siempre permanece como no asignado.

El operador *GENI* es una extensión del criterio de Reposicionamiento de nodos en el que un cliente también puede ser insertado entre dos nodos de una ruta a la que esté cercano., incluso si estos dos nodos no son consecutivos.

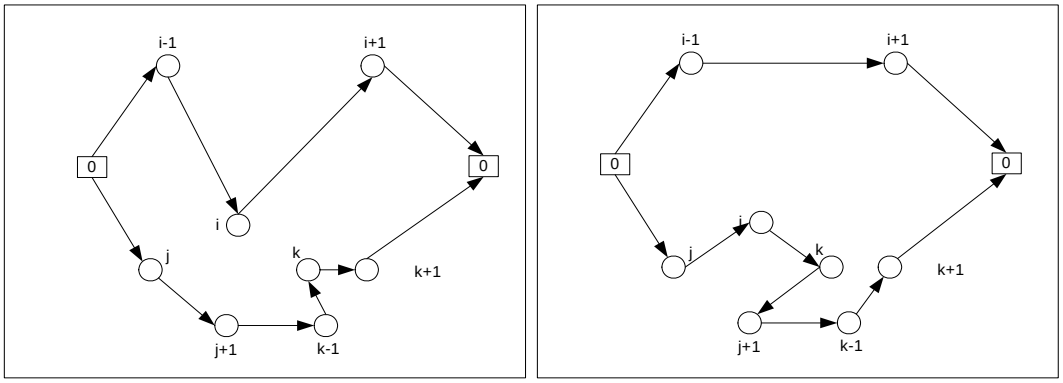


Figura 2.20 Operador de intercambio tipo *GENI*. El cliente i de la ruta superior se inserta en la ruta inferior entre los clientes *j* y *k* más cercanos, a través de la adición de los arcos *(j, i)* y *(i, k).* Dado que *j* y *k* no son consecutivos, es necesario reordenar la ruta inferior. En este ejemplo la ruta resultante se obtiene eliminando los arcos *(j, j+1)* y *(k-1, k),* y posicionando la secuencia *{j+1, ..., k-1}*

Para reducir la complejidad de la búsqueda algunos autores proponen mecanismos de limitación del vecindario. Por ejemplo, García et al. (1994) solamente permiten movimientos que impliquen arcos que están cercanos en distancia.

Taillard et al. (1997) descomponen las soluciones en una colección de conjuntos disjuntos de rutas a través de la utilización del ángulo polar asociado al centro de gravedad de cada

ruta. La búsqueda tabú se aplica a continuación de forma separada a cada subconjunto. La solución final se consigue con la fusión de todas las rutas encontradas en la fase de búsqueda tabú.

Otra estrategia comúnmente utilizada para acelerar la búsqueda es la implementación paralela en múltiples procesadores. Por ejemplo, Badeau et al. (1997) aplican el enfoque de Taillard et al. (1997) utilizando una implementación paralela de segundo nivel. Los resultados muestran que esta paralelización no perjudica a los resultados conseguidos por este método, pero mejora sustancialmente los tiempos de computación. Otros estudios describen la implementación paralela de la búsqueda tabú, como Garcia et al. (1994), Schulze y Fahle (1999).

Por otro lado, para incrementar el espacio de las búsquedas locales, marcado siempre por las ventanas de tiempo impuestas en las restricciones, algunos autores permiten soluciones no factibles con las restricciones, en las fases intermedias de la búsqueda tabú, para de esta manera tener más opciones en la consideración de arcos individuales. Por ejemplo Cordeau et al. (2001) permite la violación de todas las restricciones del problema (carga, duración y ventanas de tiempo). Las violaciones de estas restricciones implican una penalización en la función objetivo, de forma que el ajuste de los parámetros de cada una de las restricciones se realiza de forma dinámica.

En cuanto a las estrategias de reducción del número de rutas, que suele ser el objetivo primario, es muy amplio el número de posibilidades. Por ejemplo, los algoritmos de Garcia et al. (1994) y Potvin et al. (1996) intentan mover los clientes de aquellas rutas con reducido número de nodos, hacia otras rutas utilizando los mecanismos de inserción de tipo Or. Igualmente Schulze y Fahle (1999) intentan eliminar las rutas que tengan como máximo tres clientes a través de su inclusión en otras rutas.

La mayor parte de las búsquedas tabú propuestas utilizan estrategias de intensificación y diversificación muy especializadas para guiar la búsqueda. Por ejemplo Rochat y Taillard (1995) proponen la denominada "*memoria adaptativa*". Esta memoria es un conjunto de rutas recogidas de las mejores soluciones encontradas durante la búsqueda. La finalidad es proporcionar nuevas soluciones de partida para la búsqueda tabú a través de la selección y combinación de rutas extraídas de la memoria. La selección de las rutas incluidas en la memoria se realiza de forma probabilística, de manera que la probabilidad de seleccionar una ruta en particular depende del valor de la solución a la que pertenece esa ruta. Las rutas seleccionadas son a continuación mejoradas por la búsqueda tabú e insertadas a continuación en la memoria adaptativa. Posteriormente Taillard et al. (1997) utilizan la misma estrategia para tratar el problema de VRPTW con ventanas de tiempo flexibles. En este tipo de problemas la llegada a los nodos en consideración se puede realizar con un retraso que posteriormente es penalizado e incluido en la función objetivo. Taillard et al. (1997) también diversifican la búsqueda penalizando aquellos movimientos más frecuentes e intensificando la búsqueda y reordenando los clientes a través del mecanismo de inserción *I1* de Solomon. Chiang y Russell (1997), Schulze y Fahle (1999) y Cordeau et al. (2001) utilizan una estrategia similar para la diversificación, pero en

Chiang y Russell (1997) la intensificación se utiliza para reducir el tiempo de espera a través de la prohibición de intercambio de ciertos clientes entre rutas. Schulze y Fahle (1999) también proponen una estrategia similar a la memoria adaptativa, en donde todas las rutas generadas en la búsqueda tabú se recogen en listas. El finalizar las fases de optimización local, la peor solución se sustituye por una nueva creada a través de la resolución del problema de generación de conjuntos con las rutas integradas en la lista utilizando el método heurístico de relajación lagrangiana de Beasley (1990).

Carlton (1995) y Chiang y Russell (1997) prueban una búsqueda tabú reactiva que ajusta dinámicamente sus parámetros en función del momento actual de la búsqueda para evitar tanto los posibles bucles como búsquedas excesivamente restringidas. De forma más precisa, el tamaño de la lista tabú se incrementa si existen soluciones idénticas, que ocurren muy frecuentemente, y se reduce si no es posible encontrar soluciones factibles. Tan et al. (2000) diversifican la búsqueda cada vez que se encuentran con un óptimo local a través de la ejecución de un determinado número de λ intercambios aleatorios combinados con el mecanismo 2-opt*. En todo momento se mantiene una lista de candidatos para registrar las mejores soluciones encontradas durante el proceso de búsqueda. Estas soluciones "é*lite*" se utilizan después como punto de partida para la intensificación. Lau et al. (2000) presentan una técnica de diversificación basada en restricciones donde el VRPTW se modeliza como un problema de satisfacción de restricciones lineales que se resuelve a través de un algoritmo sencillo de búsqueda local.

Finalmente, muchos autores presentan diversas técnicas de post-optimización. Por ejemplo, Rochat y Taillard (1995) resuelven de forma exacta el problema de particionado de los nodos totales del problema en áreas más pequeñas para generar las rutas, utilizando las rutas de la memoria adaptativa para devolver la mejor solución posible. Taillard et al. (1997) aplican una adaptación del heurístico *GENIUS* (Gendrau et al, 1992) para las ventanas de tiempo para cada ruta individual al final de la solución. Igualmente, en Cordeau et al. (2001) la mejor solución identificada después de *n* iteraciones, se post-optimiza a través de la aplicación a cada ruta individual de un heurístico especializado en la resolución del TSPTW (Gendrau et al., 1998).

2.15.2 Recocido Simulado

Chiang y Russell (1996) desarrollan el primer enfoque de recocido simulado para el VRPTW. Los fundamentos de esta técnica de aproximación consistían en la relajación estocástica, asociada a la analogía existente en el proceso de recocido de sólidos, donde un sólido se calienta a altas temperaturas y gradualmente se va enfriando para que cristalice en una determinada configuración.
Aarts et al. (1997) combinan el procedimiento de recocido simulado con el enfoque de construcción paralela de rutas propuesto por Russell (1995), incorporando una serie de procedimientos de mejora durante la construcción. Además se implementan dos estructuras de vecindario, el mecanismo de λ-intercambios *(λ=1),* y el proceso de intercambio de *k*-nodos de Christofides y Beasley (1984), basados siempre en la aceptación de la mejor estrategia de entre estas dos. También se considera la posible

mejoría del recocido simulado a través de la incorporación de una memoria a corto plazo a través de una lista tabú dinámica.

Thangiah et al. (1994) desarrollan un conjunto de algoritmos metaheurísticos basados en un enfoque de dos fases. En la primera fase, se crea una solución inicial, bien a través del procedimiento de Inserción más Económica, o bien a través del algoritmo genético *GIDEON* de sectorización de clientes, de Thangiah (1995a). La segunda fase aplica alguno de los siguientes procedimientos de búsqueda basados en el mecanismo de λ-intercambios: un procedimiento de búsqueda descendente a través de una estrategia del primer mejor, o el mejor global; un recocido simulado con esquema no monótono de enfriamiento; o un híbrido entre recocido simulado y búsqueda tabú. En este caso se combina el recocido simulado con la búsqueda tabú para determinar qué movimientos ejecuta el recocido simulado de entre la lista de candidatos de la búsqueda tabú.

Tan et al. (2000) desarrollan un método de recocido simulado muy rápido consistente en intercambios de dos nodos simultáneos basado en la aceptación de la mejor estrategia y un enfoque de enfriamiento monótono. Después de alcanzada la temperatura final, se producen nuevas iniciaciones según las temperaturas inicial y mejor. La solución inicial utilizada se obtiene a través de una modificación del algoritmo de Thangiah et al. (1994).

Li et al. (2001) proponen un recocido simulado combinado con la búsqueda tabú. Las soluciones iniciales se generan a través de los heurísticos de Inserción y de Barrido de Solomon (1987). Se contemplan igualmente tres mecanismos de búsqueda local fundamentados en el intercambio de segmentos de clientes inter e intra ruta. El recocido simulado comienza desde la mejor solución encontrada en diferentes ocasiones. El heurístico de inserción de Solomon se utiliza para reducir el número de rutas a través de inserciones de los clientes de una ruta a otras y procurando la reprogramación de los clientes de cada ruta. Por último, la búsqueda se diversifica a través de la implementación de algunos cambios aleatorios de segmentos de clientes.

2.15.3 Algoritmos genéticos.

Tal y como se comenta en los apartados anteriores para el caso del TSP y del VRP, el algoritmo genético es un método heurístico de búsqueda adaptativa basado en la evolución genética de una población.

Blanton y Wainwright (1993) fueron los primeros en aplicar los algoritmos genéticos para la resolución del VRPTW. Para ello hibridizaron un algoritmo genético con un algoritmo de tipo *greedy*. Bajo este enfoque, el algoritmo genético buscaba una buena programación de los clientes, mientras que el algoritmo de construcción generaba las soluciones factibles de cada ruta.

En los últimos años, múltiples artículos profundizan en la aplicación de los algoritmos genéticos para encontrar buenas soluciones. La mayor parte de estos artículos aplican combinaciones de los algoritmos genéticos con la aplicación de algún heurístico de construcción (Blanton y Wainwright, 1993; Berger et al., 1998; y Bräysis, 1999a y

1999b) de búsqueda local (Thangiah 1995a y 1995b; Thangiah et al., 1995; Potvin y Bengio, 1996; Zhu 2000 y Bräysis et al., 2000) así como otros metaheurísticos como la búsqueda tabú (Wee Kit, 2001) y sistemas de colonias de hormigas (Berger et al., 2001).

5.1.1.1 Generación de la población inicial

Homberger y Gehring (1999) y Gehring y Homberger (1999 y 2001) presentan dos metaheurísticos evolutivos para el VRPTW. La representación individual de los cromosomas incluye un vector de los denominados "parámetros de la estrategia" además del vector propio de la solución de manera que ambos componentes evolucionan a través de los operadores de mutación y recombinación. En la aplicación de este método al VRPTW, estos parámetros de la estrategia se refieren a la frecuencia con la que un mecanismo de búsqueda local es seleccionado y a un parámetro binario utilizado para alternar la búsqueda para minimizar bien el número total de vehículos o bien la distancia total. La selección de los padres se realiza de forma aleatoria, y solamente un descendiente se crea a través de la recombinación de los padres. Se crea así un número de descendientes $\lambda > \mu$, donde μ es el tamaño de la población. Al final, se utilizan los valores de ajuste para seleccionar los μ descendientes para la siguiente población. En Gehring y Homberger (1999 y 2001), el algoritmo evolutivo se combina con una búsqueda tabú para minimizar la distancia total. Estos autores también desarrollan un conjunto de problemas tipo más grandes, pero fundamentados en los problemas de Solomon (1987).

En la mayor parte de las aplicaciones de los algoritmos genéticos al problema del VRPTW, se utilizan directamente las soluciones propuestas por otros algoritmos de construcción de rutas, evitándose la codificación de estos algoritmos. Para ello se tratan largas cadenas de números enteros, representativos de las soluciones del problema, y que se asocian a las cadenas de ADN. Por ejemplo, en Thangiah (1995a y 1995b) y Thangiah et al. (1995) se utiliza la codificación tradicional de cadenas de bits, donde cada cromosoma representa un conjunto de posibles rutas generadas a través de una estrategia de cluster primero y ruta después.

La población inicial se suele crear bien de forma aleatoria, o bien utilizando modificaciones sobre los algoritmos de construcción más conocidos. Un ejemplo de estrategia aleatoria se puede encontrar en Blanton et al. (1993) y Rahoual et al. (2001). Thangiah (1995a y 1995b) y Thangiah et al. (1995) agrupan aleatoriamente los clientes, y después utilizan el algoritmo de Inserción más Económica de Golden y Stewart (1985) para asignar los clientes a las rutas, dentro de cada grupo. Homberger y Gehring (1999) y Gehring y Homberger (1999 y 2001) utilizan una modificación del método de los ahorros de Clarke y Wright (1964), donde el nodo seleccionado se obtiene aleatoriamente de la lista de los ahorros provocados. Berger et al. (2001) modifican la población aleatoriamente generada con los mecanismos de λ-intercambio y un procedimiento de reiniciación basado en el procedimiento de inserción de Liu y Shen (1999), con la finalidad de crear una población de soluciones donde el número de vehículos iguale al menor encontrado. El método de inserción de Solomon se utiliza en Potvin y Bengio (1996), Bräysis (1999a y 1999b), Zhu (2000) y Tan et al. (2001). Berger et al. (1998) utilizan el heurístico de Solomon del vecino más cercano, mientras que Bräysis et al.

(2000) utilizan una modificación de los algoritmos genéticos propuestos en Berger et al. (1998) y Bräysis (1999a y 1999b) para generar una solución inicial para el algoritmo evolutivo. Pocos autores utilizan más de una población, como es el caso de Berger et al. (1998) y Berger et al. (2001). Por ejemplo, este último implica dos poblaciones en paralelo. La primera población se utiliza para minimizar la distancia total recorrida, y la segunda población intenta minimizar las violaciones de las restricciones de las ventanas de tiempo.

Los *valores de ajuste* que suponen la función objetivo del problema se fundamentan en los costes totales de las rutas, que deriva del número de rutas, distancia total recorrida y duración. Bräysis (1999a y 1999b) también considera el tiempo de espera, y en Blanton y Wainwright (1993) el valor de ajuste es el número de clientes no servidos en el caso de soluciones no factibles. Homberger y Gehring (1999) y Gehring y Homberger (1999 y 2001) también consideran la facilidad con la que se puede eliminar la ruta más corta de la solución (en términos del número de clientes de la misma), además del número de rutas y la distancia total. En Rahoual et al. (2001) y Berger et al. (2001), la evaluación de los individuos se basa en la suma ponderada de los objetivos relativos a la violación de restricciones, número total de vehículos y distancia total.

5.1.1.2 Selección de los padres

El enfoque más típico de selección de un par de individuos para ser padres consiste en el conocido método de la *ruleta*. En este enfoque estocástico, la probabilidad de seleccionar un individuo es proporcional a su valor de ajuste. Para más detalles, ver Goldberg (1989). Bräysis (1999b), Zhu (2000) y Tan et al. (2001) utilizan el enfoque del *torneo*, donde la idea principal es la de aplicar esta selección aleatoria dos veces para escoger el mejor de los dos individuos seleccionados. En Wee Kit et al. (2001), Homberger y Gehring (1999) y Gehring y Homberger (1999 y 2001) los padres se seleccionan aleatoriamente. Por último Potvin y Bengio (1996) utilizan un enfoque de lista jerarquizada donde la probabilidad de seleccionar un individuo se basa en la posición que este ocupa en la lista.

5.1.1.3 Recombinación o reproducción

La *recombinación* es la parte más importante de los algoritmos genéticos. El cruce tradicional de 2 puntos, donde se intercambian una porción aleatoriamente seleccionada de la cadena entre los cromosomas se utiliza en Thangiah (1995a y 1995b) y Thangiah et al. (1995), mientras que en Tan et al. (2001) se utiliza el conocido cruce *PMX*. La idea principal del cruce *PMX* consiste en la elección al azar de dos puntos de corte, a partir de los que se establecen una serie de permutas en el segundo padre. Blanton y Wainwright (1993) utilizan los operadores clásicos basados en una relación de precedencia entre los genes de un cromosoma, al igual que en Homberger y Gehring (1999). Le Bouthillier et al. (2001) utilizan un operador de fusión que persigue la unión de dos padres para conseguir un descendiente a través de la selección de elementos de los padres según sus valores de ajuste. En Homberger y Gehring (1999) las secuencias de nodos generadas por los procesos de mutación aleatoria son modificadas por el operador de cruce.

En el campo de la problemática del VRPTW la mayor parte de los autores proponen métodos heurísticos especializados, en vez de los operadores tradicionales. Potvin y Bengio (1996) proponen un operador de cruce entre rutas para el orden de los nodos. Este operador, basado en la secuenciación de los nodos, primero selecciona de forma aleatoria un vínculo en cada una de las soluciones padres y después los clientes, que siguen siendo servidos por el padre P_1, se unen a los clientes que dejan de ser servidos por P_2, y viceversa.

El operador basado en las rutas sustituye una ruta de la solución padre P_1 por otra ruta de la solución padre P_2. En Berger et al. (1998) y Berger et al. (2001), primero se lleva a cabo un procedimiento de retirada de algunos clientes clave de una forma similar a cómo lo hace Shaw (1998). Después se aplica localmente un procedimiento de inserción al estilo de Solomon (1987) y Liu y Shen (1999), para reconstruir la solución inicial.

El primer operador de Wee Kit et al. (2001) intenta modificar el orden de los clientes en el primer padre con la finalidad de crear pares de clientes consecutivos de acuerdo con el segundo padre. El segundo operador de cruce intenta conjuntar las características comunes del las soluciones padre para generar descendientes a través de cambios en el procedimiento de selección de semillas y la función de costes de un heurístico que mantiene muchas similitudes con el de Solomon. (1987).

5.1.1.4 Mutación

La *mutación* generalmente se considera como una estrategia secundaria, y su finalidad consiste en huir de los óptimos locales. Sin embargo en el algoritmo metaheurístico de Homberger y Gehring (1999) y Gehring y Homberger (1999 y 2001), la búsqueda es dirigida por el procedimiento de mutación, que se fundamenta en los operadores clásicos de 2-opt*, Or-opt y λ-intercambios. En Potvin y Bengio (1996), Bräysis et al. (2000), Wee Kit et al. (2001), Le Bouthillier (2001) y Rahoual et al. (2001), la mutación se fundamenta por completo en los operadores clásicos de búsqueda local. Potvin y Bengio (1996), Berger et al. (1998), Bräysis (1999a y 1999b), Homberger y Gehring (1999), Gehring y Homberger (1999 y 2001) y Berger et al. (2001), utilizan la mutación para reducir el número de rutas a través de la utilización del procedimiento de intercambio de Or, el algoritmo de inserción de Solomon o a través de la implementación de uno o varios movimientos de reposicionamiento de nodos. Berger et al. (1998) utilizan la mutación para reorganizar las rutas a través del procedimiento del vecino más cercano de Solomon (1987). En Zhu (2000), la mutación se utiliza para dar la vuelta al orden o secuencia de nodos en un tramo de una ruta. Berger et al. (2001) presentan cinco operadores de mutación fundamentados en las técnicas de Gambardella et al. (1999) de colonias de hormigas. Estos seis operadores de mutación son el LNS de Shaw (1998), los λ-intercambios, el intercambio de clientes servidos con retrasos, la eliminación de las rutas más cortas utilizando el procedimiento de Liu y Shen (1999) y la reordenación de los clientes de una ruta a través del procedimiento de Solomon (1987).

5.1.1.5 Intensificación

En algunos artículos también se describen las técnicas más recientes de *intensificación* que se aplican conjuntamente con los algoritmos genéticos. Por ejemplo Zhu (2000) y Tan et al. (2001) presentan una técnica de escalada, en donde una parte aleatoriamente seleccionada se mejora a través de la aplicación parcial de λ-intercambios. En Wee Kit et al. (2001) se aplica una búsqueda tabú muy sencilla basada en los vecindarios 2-opt*, el intercambio, el reposicionamiento y el 2-opt a las soluciones individuales en las primeras generaciones para intensificar la búsqueda. Al igual que en la búsqueda tabú, en muchos algoritmos genéticos se permite la violación de las restricciones para escapar de los óptimos locales. Ejemplos de estas estrategias se pueden encontrar en Blanton y Wainwright (1993), Thangiah (1995a y 1995b), Thangiah et al. (1995), Le Bouthillier et al. (2001) y Berger et al. (2001). Bräysis et al. (2000) utilizan una técnica de diversificación en donde los arcos mayores que la media son penalizados si aparecen con frecuencia en la búsqueda. Las implementaciones paralelas también son corrientes, como en Gehring y Homberger (1999) y Le Bouthillier et al. (2001).

2.15.4 Algoritmos combinados

Kontoravdis y Bard (1995) proponen un método de búsqueda adaptable en dos fases y de tipo *greedy* para el VRPTW. La fase de construcción de rutas comienza con la iniciación de una serie de rutas a través de la selección de un conjunto de clientes semilla que o bien su dispersión geográfica es máxima, o bien son los clientes con más restricciones temporales. Después de la iniciación, el algoritmo busca los puntos de inserción mejores en cada ruta, para cada cliente no asignado, y calcula un valor de penalización utilizando la función de costes de Solomon (1987). Esta penalización es la suma de las diferencias entre el menor coste de inserción para cada ruta y el mejor coste total. Se genera a continuación una lista L con los clientes no asignados, de manera que el siguiente cliente para ser asignado se selecciona de forma aleatoria de entre los clientes de la lista. Seguidamente se aplican los procedimientos de búsqueda local cada cinco iteraciones. En la fase de búsqueda local, se considera la eliminación de cada una de las rutas creadas, teniendo primeramente en cuenta aquellas con menor número de clientes. Si no es posible insertar el cliente en cuestión en ninguna ruta, entonces se plantea la retirada de algún cliente de la ruta destino, siendo insertado en otra ruta, para añadir al cliente en consideración. Finalmente se aplica un procedimiento de mejora de tipo 2-opt para mejorar la solución en términos de distancia total recorrida.

Para estimar el número de rutas mínimo, los autores presentan tres límites inferiores. El primero se obtiene de la estructura creada por las restricciones de carga subyacentes. El segundo se obtiene a través de la consideración conjunta de las restricciones de carga y las ventanas de tiempo. Por último, el tercer límite se obtiene a través de la consideración de la agrupación de ventanas de tiempo asignadas a las rutas.

5.1.1.6 Redes Neuronales

Potvin y Robillard (1995) combinan el heurístico de Inserción más Económica de Potvin y Rousseau (1993) con un modelo de redes neuronales. Para cada ruta se define un vector

de carga. Al principio todos los vectores de carga se localizan cercanos al depósito de forma aleatoria. A partir de ahí, se van seleccionando clientes de uno en uno y se calcula su distancia a cada uno de los vectores de carga. El vector de carga al que esté más cercano se actualiza añadiendo este cliente, desplazando el vector hasta este cliente. Este proceso se repite para todos los clientes, de forma que los vectores de carga se vuelven menos sensibles en cada una de las iteraciones. Al final de esta fase se seleccionan los clientes semilla de los nodos más cercanos a los vectores de carga.

Potvin, Dubé y Robillard (1996) reutilizan el enfoque anterior para seleccionar los clientes semilla del algoritmo presentado por Potvin y Rousseau (1993). En este algoritmo se necesitan tres parámetros, α_1, α_2 y μ. Las dos primeras constantes determinan la importancia de la distancia y el tiempo de viaje en la función de costes para cada uno de los clientes no asignados. La tercera constante se utiliza para controlar los ahorros en distancia. Se utiliza un algoritmo genético para encontrar los valores de estos parámetros. Se aplica un procedimiento estocástico en el proceso de selección a los valores de ajuste basado en el número de rutas y el tiempo total de las mismas de la mejor solución generada por el heurístico de inserción en paralelo. Se utiliza el clásico procedimiento de intercambio de 2 puntos para la recombinación, a través del que se permuta un segmento de nodos consecutivos de un padre a otro. Los cambios provocados pro el procedimiento de mutación se generan con una probabilidad muy baja y se aplican directamente a cada nodo de la cadena. Los resultados mejoran ligeramente las soluciones originales del algoritmo inicialmente propuesto por estos autores.

5.1.1.7 Algoritmos de Negociación

Bachem et al. (1996) proponen un heurístico de mejora basado en los procesos de negociación. La partición de los clientes en rutas se determina a través de la búsqueda de las mejores asignaciones en un gráfico bipartito denominado por los autores "*grafo de negociación*". Los nodos se incluyen, o bien en la zona de posibles inserciones (*compra*), o bien en la de eliminaciones (*venta*). Los arcos representan posibles intercambios y el peso de cada arco es la ganancia correspondiente a cada acción. Por lo tanto, cada asignación del grafo de negociación se corresponde a los posibles intercambios entre clientes. En cada iteración las rutas son valoradas tras escoger alguna permuta al azar. Para cada ruta se selecciona una acción, bien de compra o bien de venta, de manera que todas las asignaciones posibles son evaluadas, y finalmente se selecciona la mejor. Este enfoque permite la violación de restricciones, incurriendo en los correspondientes factores de penalización, así como transacciones con signos negativos, de manera que se incurre en el deterioro de la solución. Por ello también se incluye una lista tabú para prevenir bucles en la negociación. Este enfoque se implementó a través de dos tipos de paralelizaciones: en la primera cada ruta se desarrolla en un procesador que establece las decisiones de compra y venta, mientras que en la segunda se reparten equitativamente el número de rutas entre todos los procesadores.

5.1.1.8 Búsqueda Local Guiada y Búsqueda Local Rápida

Kilby et al. (1999) introducen la Búsqueda Local Guiada para el VRPTW. Este procedimiento consiste en una técnica basada en los procedimientos de memoria de

Vouduris (1997) y Vouduris y Tsang (1998). Su operativa comienza a través del incremento en la función de costes con una penalización derivada de lo cercanos que estén los movimientos propuestos a los movimientos anteriormente considerados por el algoritmo y resultantes en mínimos locales. En este sentido los autores proponen la penalización de los arcos de unión entre cada par de nodos según el número de veces que aparezcan en las soluciones previamente generadas por el algoritmo. Cuanto más frecuentes sean estos movimientos, menos probabilidad tendrán de que sean penalizados en la solución final.

En la solución inicial no se asignan las visitas a ninguna ruta. Se asocia una penalización a la no visita de los nodos, y a partir de ahí el algoritmo comienza a plantear las diferentes opciones de visita. Se utiliza un procedimiento de minimización de costes a través de la implementación de cuatro mecanismos de búsqueda local: 2-opt, intercambio, reposicionamiento y 2-opt*, con la estrategia de aceptación del mejor de los cuatro.

Riise y Stølevik (1999) analizan la Búsqueda Local Guiada y la Búsqueda Local Rápida combinando los mecanismos de reposicionamiento simple de clientes. Tanto las estrategias de aceptación del primero aceptado, como del mejor aceptado, son valoradas dentro de los movimientos propuestos. Al igual que en Kilby et al. (1999) los arcos se seleccionan para ser penalizados. La idea principal de la Búsqueda Local Rápida es la de desagregar el vecindario de la solución y asociar una característica a cada uno de los subvecindarios resultantes. Cada subvecindario consiste en todas las soluciones posibles dentro de la solución global inicial, derivadas de la aplicación de los mecanismos de movimiento que se recogen en el algoritmo, de forma que la característica asociada (arco) se elimina de la solución. La búsqueda local Rápida se centra en los movimientos que eliminen arcos de la solución original. La solución inicial se genera a través de la inserción de clientes de uno en uno y de forma aleatoria.

5.1.1.9 Programación Restringida

De Backer et al. (2000) investigan las técnicas iterativas de mejora dentro de un enfoque de programación de restricciones. Estas técnicas de mejora se implementan conjuntamente con la búsqueda tabú o el recocido simulado para evitar los bucles en la búsqueda. El sistema de programación restringida se utiliza únicamente como espina dorsal para comprobar la validez de las soluciones y para acelerar el chequeo de la validez de las mejoras propuestas por los mecanismos de mejora. Se utilizan cuatro mecanismos de mejora en las búsquedas locales: 2-opt, reposicionamiento, intercambio, y 2-opt*, dentro de una estrategia de aceptación del mejor aceptado. La búsqueda tabú seleccionada es muy sencilla: incorpora dos únicas listas para almacenar los arcos recientemente retirados y los arcos insertados. La implementación de la Búsqueda Local Guiada es muy similar a la utilizada por Kilby et al. (1999). La solución inicial se genera a través del método de los ahorros de Clarke y Wright (1964), seguido de un método descendente hacia el óptimo local.

5.1.1.10 Algoritmos de Colonias de Hormigas

Gambardella et al. (1999) utilizan el enfoque fundamentado en la optimización por colonias de hormigas cooperativas. Los sistemas de hormigas están inspirados en las colonias reales de hormigas que buscan comida. En su búsqueda, las hormigas marcan los caminos que recorren mediante el depósito de una esencia aromática llamada *feromona*. La cantidad de feromona depositada en el camino depende de la longitud del camino y de la calidad de la fuente de comida (frecuencia de uso). Esta feromona proporciona información a otras hormigas que son atraídas por estos buenos caminos. Los autores utilizan dos colonias, la primera se utiliza para minimizar el número de vehículos utilizados, mientras que la segunda se utiliza para minimizar la distancia total recorrida. Para cada arco se utilizan dos medidas, la atracción N_{ij} y el camino de feromonas T_{ij}. Las rutas se construyen utilizando el algoritmo del vecino más cercano, pero con reglas probabilísticas, es decir que el cliente siguiente a ser añadido a una ruta no siempre es el mejor de acuerdo con su N_{ij} y su T_{ij}. Los caminos de feromonas se actualizan tanto a nivel local como a nivel global. El efecto de la actualización local consiste en la actualización del nivel de atracción de los arcos: cada vez que una hormiga utiliza un arco, la cantidad de feromona de ese arco disminuye de manera que ese arco se muestra menos atractivo. Por otro lado, la actualización global obedece a que la búsqueda se intensifica en el vecindario de la mejor solución encontrada. Cada hormiga artificial construye una solución factible por separado para intensificar la búsqueda y la atracción N_{ij} se calcula tomando en cuenta la distancia entre nodos, la ventana de tiempo del cliente considerado y el número de veces que el cliente no ha formado parte de la solución. Además, se aplica por último los procedimientos de mejora de intercambios de tipo *CROSS* de Taillard et al. (1997).

5.1.1.11 Algoritmos de Segregación de Cadenas

Caseau et al. (1999b) proponen un híbrido entre muchas técnicas, como la búsqueda de discrepancias limitada, la búsqueda en vecindarios amplios, las eyecciones de árboles y las eyección de cadenas. La solución inicial se construye utilizando un enfoque similar al de Caseau y Laburthe (1999a). Las eyecciones de cadenas y las eyecciones de árboles se utilizan para reposicionar los clientes más costosos en otras rutas. El principio básico de la eyección de árboles es el de retirar más de un cliente de una ruta, a la vez que se inserta un solo cliente en la ruta original. La búsqueda en vecindarios amplios se utiliza como en Shaw (1998). Por último los autores desarrollan un álgebra nueva para definir y desarrollar estos operadores utilizando diferentes valores de parámetros. Además proponen una nueva técnica de ajuste automático de parámetros a través de un proceso de aprendizaje de forma que para cada nueva combinación, el algoritmo se ajusta de forma automática. En este sentido los autores demuestran que el ajuste automático proporciona mejores resultados que el ajuste manual y con muchos menos esfuerzos.

5.1.1.12 Otros algoritmos

Liu y Shen (1999) proponen un método metaheurístico de dos fases basado en una estructura de vecindario, enfocado en la relación entre nodos y rutas. En la fase de construcción, las rutas se forman de una manera paralelizada a través de la repetida

estimación del límite inferior de los clientes no asignados. Los clientes semilla se seleccionan de acuerdo con las diferencias en las ventanas de tiempo, o bien a su dispersión geográfica máxima, con respecto al par de clientes entre los que existan un mayor número de clientes en camino. El segundo paso consiste en utilizar estas rutas parciales para servir los clientes no asignados hasta que ya no existan posibles inserciones de los mismos. Si no es posible asignar un cliente a ninguna ruta, entonces se aplican cinco tipos de mecanismos: estas operaciones incluyen la inserción de uno o dos nodos en otras rutas a través de la eyección de una cadena y su posterior reorganización. Además se genera un nuevo conjunto de clientes no asignados a través del intercambio entre clientes asignados y no asignados. Durante la fase de construcción se eliminan aquellas rutas con cargas reducidas. Una vez que se obtiene una solución factible, entonces se aplican λ-intercambios e inserciones simples dentro de las rutas para mejorar la solución obtenida. Por último, se utilizan reinserciones intra ruta para escapar de los posibles óptimos locales.

Bräysis (2001c) presenta un nuevo método determinista en cuatro fases basado en una modificación de la búsqueda en vecindario variable de Mladenovic y Hansen (1997). En la primera fase, se genera una primera solución de partida a través de un algoritmo heurístico de construcción de rutas, que se fundamenta en los estudios de Solomon (1987) y Russell (1995). Las rutas se generan de forma secuencial, y después de que k nodos hayan sido insertados a una ruta, entonces se reordena utilizando el procedimiento de intercambio de Or. En la segunda fase se aplica un procedimiento especial de eliminación de rutas basado en un nuevo sistema de eyección de cadenas para minimizar el número de rutas. En la tercera fase, las soluciones creadas se mejoran en términos de distancia utilizando una oscilación en el vecindario a través de cuatro nuevos procedimientos de mejora. Estos procedimientos se fundamentan en los intercambios de tipo *CROSS* de Taillard et al. (1997), y los procedimientos de Inserción más Económica. En la cuarta fase, la función objetivo utilizada en la búsqueda local se modifica para considerar también los tiempos de espera y huir así de los óptimos locales.

Benyahia y Potvin (1995) utilizan un enfoque similar al de los algoritmos genéticos para optimizar los parámetros de las versiones secuencial y en paralelo del algoritmo de Solomon (1987). Sin embargo, aquí los clientes semilla se seleccionan como en el algoritmo original y no como en el caso de las redes neuronales. Los autores incluyen medidas adicionales de coste, donde se contemplan los tiempos perdidos y las esperas; comparativa entre los ahorros generados en una inserción y la opción de crear una nueva ruta; y un ratio entre la distancia adicional recorrida en una inserción y la distancia original entre cada par de clientes consecutivos.

5.1.1.13 Optimización Interactiva

Anderson et al. (2000) proponen un enfoque interactivo para el VRPTW. La idea fundamental consiste en la utilización de un algoritmo ascendente basado en el reposicionamiento de n clientes para encontrar un mínimo local. Después, utilizando técnicas de visualización e interacción, el usuario identifica aquellas áreas donde desea que la búsqueda se intensifique, ayudando a que el algoritmo escape de los óptimos

locales. Por ejemplo, la función de evaluación puede ser modificada durante la búsqueda por el usuario, de manera que éste define nuevas prioridades a determinados clientes para forzar algunos movimientos, así como escoger entre las estrategias del primero aceptado o el mejor aceptado. Por último se aplica un algoritmo de agrupación y acotación para optimizar cada ruta por separado.

6 Aplicación de los métodos de resolución a casos reales

El objetivo de este apartado es poner de manifiesto la importancia de la aplicación de las técnicas estudiadas en los primeros apartados, referentes a la optimización de rutas, a casos reales donde establecido un problema determinado del sistema logístico de una compañía, se desarrollan las soluciones de sistemas de información, generalmente del tipo de sistemas expertos, para solventar o facilitar la optimización del sistema.

Fisher, Greenfield, Jaikumar y Lester (1982) tratan la aplicación del programa informático *ROVER* para la reducción de costes en *Du Pont, Inc*. Se hace una descripción detallada del problema, aunque no del programa ni del algoritmo propuesto. Se logra reducir los costes de distribución en un 15 %. El problema trata de asignar 1.500 clientes en 50 rutas. La descripción es bastante general, sin descender a los detalles del proyecto.

Van Vliet, Boender y Kan (1992) tratan una aplicación de optimización de rutas en el ámbito de distribución de azúcar por parte de *Suiker Unie*, la cooperativa holandesa de agricultores que comercializan cerca del 60% de la remolacha azucarera del mercado holandés. Se trata de un problema VRPTW de reparto a granel o en masa con varios depósitos. Los vehículos reparten cargas completas a cada cliente y pueden ser recargados en cualquiera de los cinco depósitos disponibles para ello. La recarga de vehículos es intensiva en tiempo y por ello, en cada depósito no se pueden cargar dos vehículos de manera simultánea. Para la resolución del problema se formuló un programa matemático que ha resuelto el problema de cálculo de rutas de vehículos y además ha mejorado la eficiencia y la calidad del proceso de planificación. Introduce la valoración de restricciones de ventanas de tiempo blandas y duras.

Erkut, MacLean (1992) comentan y describen los problemas de transporte y distribución para el *Energy Efficiency Branch* perteneciente al *Alberta Department of Energy* (Canadá). Para ello se construyó un programa informático de control integral del transporte: selección de vehículos, mantenimiento de vehículos, entrenamiento de conductores y rutas de vehículos. El estudio aquí desarrollado permitió un ahorro anual del 10% en los costes de combustible. Para algunos vehículos de la flota se logró un ahorro superior al 20%. No obstante, este ahorro se logró no sólo mediante un diseño óptimo de rutas de vehículos, sino también mediante una mejora en la forma de conducción de los conductores, elección de vehículos, motores de consumo menor, etc. Respecto a los problemas de rutas, los autores resuelven tres tipos de ellos:

- El primer problema resuelve el cálculo de una única ruta para un único vehículo que visita a los clientes una sola vez.
- El segundo plantea una vuelta de recogida y distribución de mercancías para un único vehículo.

- El tercero establece rutas para múltiples vehículos que tienen restricciones de capacidad, de longitud de ruta y de número de clientes visitados en cada vuelta (tour).

Braklow, Graham, Hassler, Peck y Powell (1992), tratan la aplicación de sistemas de optimización de rutas de transporte en la empresa *Yellow Freight System*, dedicada a distribuir mercancías, con más de 15 millones de entregas en una red de 630 nodos. Con el crecimiento de la compañía, se hizo necesario el estudio de optimización de las rutas de los vehículos. Para ello se diseñó un programa de optimización de rutas a gran escala llamado *SYSNET*. Este programa se ha revelado como extraordinariamente útil para la mejora de los tiempos de distribución de mercancías y en la fiabilidad del servicio. El artículo presenta una gran cantidad de detalles en la aplicación del programa (Braklow, 1993).

Otro de los casos contemplados por la literatura es el de la empresa *CLAS*, una central lechera de Asturias, al norte de España. De manera similar a los trabajos de Golden y Wasil, (1987), Adenso et al. (1998) plantean un sistema de apoyo a la toma de decisiones, basado en una jerarquía de las mismas. Tratan el Sistema de Información Logística de manera jerárquica, optimizando en primer lugar los niveles de decisión más elevados hasta descender hasta el nivel más bajo. Los autores en este caso distribuyen la estructura jerárquica en cinco niveles:

1. Número de visitas a cada cliente por semana por un vendedor cualquiera.
2. Distribución de los clientes entre los promotores de ventas. En este sentido se intentó adjudicar clientes a los promotores de manera equitativa, por reparto uniforme en el territorio, por volumen de compras de los mismos y por homogeneidad en las asignaciones de los distintos días de la semana.
3. Distribuir los clientes del promotor entre los comerciales a su cargo.
4. Distribuir los clientes del comercial entre los días de la semana, atendiendo al número de visitas realizadas.
5. Diseño de la ruta, incluyendo el orden de visitas para cada día de la semana y para cada comercial.

El sistema se implantó empleando datos geográficos reales, donde los puntos de demanda tienen unas coordenadas de latitud y longitud determinadas y las distancias son reales, dado que las distancias euclídeas pierden realismo para casos donde existen impedimentos geográficos, como montañas, etc. De acuerdo con Potvin, Lapalme y Rousseau, (1994), un sistema logístico ha de poder mantener una red de carreteras que permita cinco beneficios clave:

- Habilidad para generar matrices de distancias
- Editar redes de carreteras reales
- Actualizar las carreteras
- Relocalizar el depósito
- Generar caminos o rutas exactas.

La solución de los problemas números 2, 3 y 4 se obtiene a través de la creación de clusters, lo cual es fácilmente realizable empleando heurísticos. La solución al TSP se obtiene con el planteamiento de 2-opt, tratando de buscar las mejores combinaciones posibles hasta que ninguna mejora se pueda alcanzar.

En cuanto a la implementación del sistema, cabe mencionar la particularidad de que es un sistema interactivo de optimización donde el usuario, generalmente poco experimentado en programas informáticos, tiene la capacidad de modificar manualmente las soluciones propuestas por el sistema. Además se destaca la necesidad de que sea un sistema de fácil utilización y manejo por el usuario, basado en iconos y ventanas para su mayor sencillez y que permita la visualización de los resultados (Bodin y Levy, 1994).

A la hora de planificar las necesidades de información de los nodos, se optó por los códigos postales como la medida más exacta de localización de los clientes. De esta manera las rutas se hacen utilizando una secuencia de códigos postales visitados de cada vez, muchos de los cuales agrupan a más de un cliente.

Generalmente un planteamiento como este implica la utilización masiva de los niveles jerárquicos inferiores y una menor utilización de los modelos de ámbito superior, los cuáles se nutren de la información de toda la operativa para una vez al año reasignar las zonas a los promotores.

Weigel y Cao (1999) tratan de la aplicación de técnicas de investigación operativa conjuntamente con técnicas de representación gráfica (GIS) de las rutas optimizadas para el caso de la empresa *Sears*. En cuanto a la resolución de los problemas de optimización, se opta por una heurístico basado en cluster primero–rutas segundo y además se incorporan multitud de nuevas restricciones, cuyo tratamiento va a ser más manual que algorítmico, lo cual refleja la necesidad de desarrollar algoritmos que permitan una cierta interactuación con el motor de solución del sistema. Cabe destacar sus comentarios iniciales en cuanto al planteamiento inicial del problema, ya que describe con todo detalle cuál es la situación de *Sears* en cuanto a su sistema logístico y cuáles son sus problemas básicos.

Fernández de Córdoba, P., García-Raffi, L.M., Mayado, A. y Sanchis, J. M. (2000) proponen en uno de los últimos trabajos publicados la aplicación de técnicas de simulación para el establecimiento de rutas logísticas en la empresa. Se trata de un artículo que desarrolla técnicas de Monte Carlo para resolver un problema de una empresa de distribución de comidas en Valencia (España). El problema se modeliza como un conjunto de once ejemplos de VRP. Para su resolución se desarrolla un heurístico de Monte Carlo que logra unos ahorros, respecto de la solución inicial de la empresa distribuidora, del 20% en distancia y del 5% en coste.

7 Conclusiones

En este capítulo se ha desarrollado una revisión del tratamiento del problema de la planificación y programación de rutas de transporte, desde los métodos más sencillos aplicados al problema del TSP, hasta los métodos más complejos en el caso del VRPTW.

Es destacable la gran dependencia de los últimos con respecto a los primeros, dado que ciertamente el problema general del VRP consiste en una generalización del problema del TSP, y que en su variante del VRPTW, incluye el criterio de temporalidad, lo que lo acerca todavía más a una situación realista de planificación. Es en esta línea del VRPTW, donde se enmarcan la mayor parte de los trabajos que buscan una aplicabilidad real de los resultados dado que mantiene una fácil analogía con la planificación de rutas en empresas de transporte. De hecho existe un elevado número de autores que han aportado grandes avances en el desarrollo de sistemas informáticos que actualmente se están comercializando, de manera que se vinculan todas las actividades de investigación, desarrollo e innovación en las mismas instituciones u organizaciones.

A pesar de los grandes avances conseguidos hasta el momento, motivados en gran parte por la implementación de técnicas de optimización matemática a través de cálculos intensivos, se ha quedado relegada a un segundo plano la línea de investigación consistente en los métodos de construcción de rutas.

En este sentido la mayor parte de autores optan por comenzar con una solución de partida mediocre, obtenida por cualquiera de los métodos de construcción de rutas existentes, mayoritariamente el *I1* de Solomon (1987), para implementar a continuación técnicas de mejora o de búsqueda intensiva para aproximarse al óptimo. Son pocos los autores que optan por intentar mejorar los mecanismos de construcción, frente a aquellos que profundizan en los mecanismos de mejora.

Es conveniente recordar que la eficacia de estos mecanismos de mejora, está condicionada por la eficiencia y eficacia de los métodos de construcción de rutas, cuyas soluciones finales constituyen las soluciones de partida de los procedimientos de mejora.

Por ello en este trabajo se ha optado por profundizar en los mecanismos de construcción de rutas, tanto para arrojar más luz sobre las reglas de construcción de rutas, como para reactivar la línea de investigación conducente a la obtención de métodos de construcción más potentes, y rápidos, a través de la implementación de búsquedas guiadas y controladas de nodos, al igual que realizaría un planificador en el departamento de planificación de una empresa de transportes.

Capítulo 3

Desarrollo del Modelo

En este capítulo se presenta y desarrolla el modelo de construcción de rutas de transporte. La finalidad no es otra que documentar las reglas de decisión que guían las diferentes fases de construcción de rutas que utiliza el modelo, procurando en todo momento mantener una cierta lógica más o menos intuitiva en la asignación de los nodos a las sucesivas rutas.

Este capítulo se ha dividido en tres grandes apartados. En primer lugar se recoge una presentación general del algoritmo en cuestión, en la que se detallan los fundamentos y principios en los que se apoyan las reglas de decisión aplicadas, y que dan lugar a las rutas de la solución. Igualmente se comparan las reglas propias del algoritmo, con los planteamientos más similares propuestos por otros autores, y más en concreto con los modelos de Solomon (1987). En segundo lugar se desarrolla la modelización matemática del algoritmo, utilizando la notación correspondiente, similar al resto de los autores involucrados en la resolución del VRPTW. En tercer lugar, se presenta un diagrama de flujo de las diferentes etapas de la construcción de rutas. Igualmente se recoge en el anexo II la programación de los procedimientos más importantes del algoritmo. La intención de la inclusión de estos procedimientos no es otra que aportar las secuencias de código originales del algoritmo para fomentar y posibilitar el desarrollo de futuras líneas de investigación. Igualmente se recogen algunos datos sobre el manejo de la aplicación desarrollada, con la finalidad de analizar su posible aplicabilidad para una posible implementación a casos reales.

1 Presentación y justificación del modelo

1.1 Definición general del algoritmo, fundamentos e hipótesis

En el presente apartado se establecen con detalle los fundamentos que subyacen del algoritmo presentado en este trabajo, así como las hipótesis en las que se basa y la operativa general del mismo.

En primer lugar cabe destacar que se trata de un algoritmo de construcción de rutas y no de postoptimización. Merece la pena recordar que existen dos tipos de algoritmos heurísticos de optimización, aquellos que permiten la construcción de rutas en un proceso guiado por una serie de reglas de decisión y en segundo lugar los consistentes en algoritmos de postoptimización.

Los primeros se caracterizan por contemplar una serie de rutinas que permiten la construcción de rutas partiendo del conjunto total de nodos. Estas rutas son construidas generalmente en un proceso sistemático de decisiones. En este sentido generalmente los trabajos encontrados en la literatura contemplan reglas que son lógicas en todo momento y que tratan de asemejarse a las reglas más intuitivas de decisión en la programación de rutas en situaciones reales.

Los algoritmos de postoptimización trabajan con un conjunto de rutas ya construidas y que no tienen por qué ser viables desde sus comienzos. Este tipo de algoritmos se caracterizan generalmente por fundamentarse en procesos de optimización matemática poco intuitivos y obedecen más a cuestiones relativas al propio problema de optimización, que a la representación de las reglas de decisión de los planificadores empresariales. Para ello se suele trabajar con conjuntos de rutas generados aleatoriamente sobre los que se aplican una serie de cambios o alternativas de manera que se busca en todo momento mejorar la solución de partida. Por ejemplo, el recocido simulado, la búsqueda tabú y los algoritmos genéticos son herramientas de cálculo intensivo de posibilidades que permiten probar un gran número de alternativas con procesos muy sencillos para mejorar la solución inicial.

Generalmente las soluciones de partida para estos algoritmos de postoptimización se obtienen de algoritmos sencillos y de baja calidad que permiten la construcción de un conjunto inicial de rutas, como por ejemplo el *Nearest Neighbor*, los procedimientos de inserción, o el método de los ahorros. De esta manera sobre las rutas conseguidas en estas primeras fases se aplican los algoritmos de postoptimización. En ocasiones los autores ni siquiera recurren a procesos de construcción de rutas propiamente dichos, sino que estas son generadas de forma aleatoria y, a partir de ahí, intentan mejorar las soluciones de originales.

Este tipo de algoritmos proporcionan soluciones mejores que los propios algoritmos de construcción de rutas. Sin embargo resultan difíceles de integrar en aplicaciones comerciales debido a las fuertes inversiones en tiempos de cálculo para conseguir estos

resultados, por lo que la aplicabilidad de este tipo de algoritmos a situaciones reales de empresas queda en entredicho. En ocasiones se utilizan redes de ordenadores para resolver problemas de pocos nodos, en plazos muy amplios de tiempo y sólo de esa manera se consiguen superar las soluciones mejores halladas hasta el momento. Si se analizase la aplicabilidad de alguna de estas técnicas a problemas reales nos encontraríamos con que las empresas deberían de afrontar fuertes inversiones en infraestructuras para poder obtener este tipo de soluciones, con el inconveniente de que en ocasiones deberían de esperar amplios plazos de tiempo para poder contar con una solución que mejorase las soluciones iniciales de partida de los algoritmos más sencillos e intuitivos.

Por eso en este trabajo se ha optado por desarrollar y analizar los algoritmos de construcción de rutas, intentado aportar más luz sobre las reglas de decisión que han de contemplar de forma que en breves espacios de tiempo, y generalmente con poca infraestructura, se puedan resolver problemas reales en los que se contemple un elevado número de nodos. La justificación de esta elección obedece a varios motivos:

- En primer lugar, cabe señalar que se trata de un trabajo enmarcado en el área de organización de empresas, por lo que se ha buscado la máxima cercanía con la problemática real de las empresas, sin perder el horizonte de la investigación científica. De esta manera se ha intentado en todo momento mantenerse cercano a los problemas que afectan a las empresas en el día a día, y cómo sería posible resolver esos problemas asumiendo todas las restricciones planteadas.
- En segundo lugar, los algoritmos de postoptimización generalmente trabajan con herramientas y técnicas de optimización que son puramente matemáticas, aunque en este caso se trate de problemáticas aplicadas a la vida real. Sin embargo en ocasiones pierden la esencia de la problemática empresarial, utilizando en exceso modelos que simplifican enormemente la realidad y que por ello hacen imposible su aplicación a situaciones reales. Además, tal y como se comentaba anteriormente, en ocasiones las exigencias en cuanto a velocidad de proceso y tiempo de cálculo de este tipo de algoritmos hacen inviable la utilización de los mismos en el ámbito empresarial.
- Por último, si bien es cierto que existe software de optimización de rutas de transporte con fines comerciales, este tipo de software es en ocasiones prohibitivo para PYMES, de manera que solamente se puede utilizar por las grandes corporaciones.

En resumen se ha optado por intentar arrojar más luz a los sistemas de decisión operativos para la optimización de rutas siguiendo el esquema tradicional del VRPTW, y para ello se ha centrado la atención en los procesos de construcción de rutas y que son en todo momento compatibles con los algoritmos de postoptimización.

En este sentido se presenta en este trabajo la construcción de un nuevo algoritmo heurístico de construcción de rutas para la resolución de problemas del tipo VRPTW, con los siguientes objetivos concretos:

- En primer lugar aportar una mayor comprensión sobre la problemática empresarial que atañe a la actividad de planificación de rutas de transporte con restricciones temporales y sus posibles soluciones a través de modelos matemáticos.
- Comparar las diferencias y similitudes de los problemas empresariales con la problemática y casuística recogida en la literatura.
- Desarrollar un conjunto de reglas de decisión lógicas que persigan la creación de rutas factibles procurando en todo momento la minimización de los costes totales de transporte, sin perder la conexión con la realidad de los problemas empresariales.
- Construir un algoritmo que facilite la implementación de estas reglas de decisión en un motor de cálculo que proporcione al usuario o planificador una herramienta útil para la toma de decisiones en breves espacios de tiempo.
- Proporcionar al algoritmo una interfaz de soporte de comunicación entre el usuario y el motor de cálculo que permita la entrada de datos, así como proporcione los resultados de una manera lógica y gráfica para facilitar al planificador la tarea de decidir.
- Proporcionar una gran flexibilidad en el algoritmo que permita se utilización para todo tipo de problemas empresariales relacionados con el VRPTW.

En concreto las características particulares de los problemas VRPTW que trata de resolver el método planteado, y sobre los cuales se aportan resultados concretos en el capítulo 4, se resumen en los siguientes puntos:

- Depósito central único, y múltiples clientes con cualquier tipo de dispersión (uniforme o clusterizados)
- Depósito sujeto a ventana de tiempo, indicativa del horizonte temporal de las rutas.
- Clientes sujetos a ventanas de tiempo no necesariamente iguales.
- Distancias euclídeas entre nodos (Fácilmente ampliable a tiempos de desplazamiento no euclídeos)
- Tiempos de servicio en cada cliente no necesariamente iguales
- Capacidades de los vehículos limitadas e iguales.
- Flota de vehículos ilimitada
- Objetivo último de la planificación: Minimizar la distancia total recorrida por todos los vehículos.

Estas características de los problemas particulares del VRPTW que trata de resolver el algoritmo, coinciden con las características de los problemas tipo planteados por Solomon (1987), por lo que servirán en este trabajo como medida comparativa de la eficiencia y eficacia del modelo desarrollado, a través de la comparación de la distancia total conseguida por los modelos de otros autores.

Sin embargo, en ningún momento se persigue la consecución de algoritmos que superen los mejores resultados encontrados hasta el momento por parte de los autores más destacados de la literatura, dado que ello implica la persecución de líneas de investigación fundamentadas en principios matemáticos o meramente computacionales, más propios de áreas de conocimiento relacionadas con la inteligencia artificial, las matemáticas o las ciencias de computación, tal y como se observa en la recopilación bibliográfica recogida en este trabajo. En ese sentido y con la finalidad de acercar la problemática del VRPTW más a la organización de empresas se ha optado por los algoritmos de construcción de rutas fundamentados en reglas lógicas de decisión, más que por algoritmos intensivos en cálculo donde las únicas reglas existentes son las derivadas de la prueba y error.

A pesar de ello, se ha establecido una comparativa de los resultados conseguidos por este nuevo método en comparación con los resultados conseguidos por otros, incluyendo tanto métodos de construcción de rutas como métodos de postoptimización e, incluso, métodos metaheurísticos basados en herramientas de cálculo intensivo. Estos resultados se presentan en el capítulo 4 relativo a las fases de experimentación y obtención de resultados.

1.1.1 Fundamentos del modelo

El modelo presentado en este trabajo se ha estructurado dentro de los algoritmos heurísticos de construcción de rutas. Por ello contempla en sus diferentes fases una serie de reglas de decisión que persiguen la construcción de rutas de forma simultánea, y no secuencial como otros métodos existentes. En este caso el algoritmo comienza la construcción de un conjunto de rutas desde el primer momento de la planificación, de forma paralela. En las siguientes líneas se recoge la explicación de estas reglas de asignación utilizadas por el algoritmo, acompañadas de ejemplos relativos al problema R103 de Solomon (1987), sobre el que se aplican estas reglas en las primeras etapas de la planificación.

Desde el primer momento se generan un total de R rutas, donde R es un parámetro establecido por el usuario, indicativo del número de rutas que desea generar desde el comienzo. Estas R rutas comienzan a partir de los R nodos semilla con momentos de servicio más críticos. Estos nodos serán aquellos cuyo servicio no se pueda retrasar debido a que tienen momentos de cierre muy tempranos, de forma que los vehículos no podrían realizar visitas intermedias. Para calcular esta criticidad se utiliza la holgura resultante entre el cierre de cada cliente (tc_i), y el tiempo necesario para llegar a él (t_{0i}).

Esta holgura se obtendría a través de $h_i=tc_i-t_{0i}$

De entre todos los nodos se selecciona aquellos R nodos con las menores holguras resultantes, tal y como se acaban de definir. Por ello el algoritmo genera una primera lista de R nodos semilla y los asigna a las correspondientes R rutas, a ser visitados por R vehículos independientes. En este sentido se contempla la primera de las reglas contenidas en el algoritmo que es la **regla de asignación**. Cada nodo semilla se asigna a una ruta diferente. En el siguiente caso del problema R103, se ha optado por establecer *9*

rutas iniciales, a través de la selección de los $R=8$ nodos con momentos de cierre más críticos, de manera que los resultados serían los siguientes.

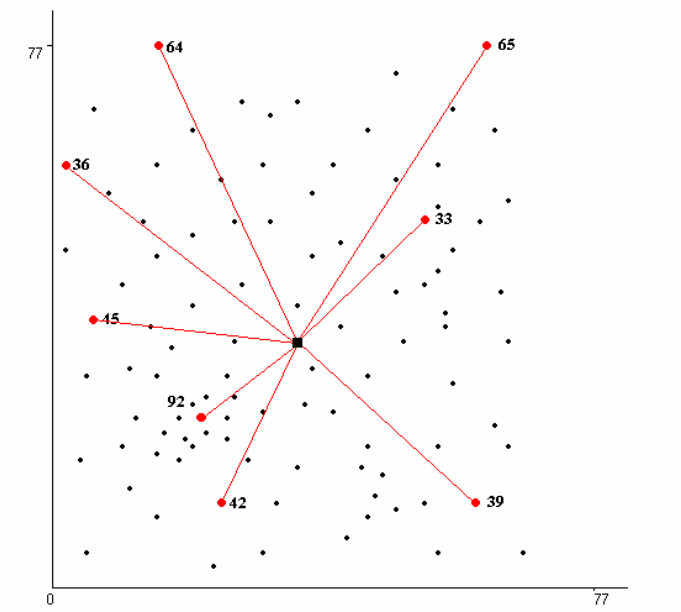


Figura 3.1 Regla de asignación. Cada uno de los *R* nodos semilla se asigna a una ruta diferente.

A continuación, y después de haber asignado estos primeros *R* nodos, se analiza cada uno de los mismos en detalle para comprobar si en las holguras resultantes de las visitas de estos nodos, es posible incluir alguna visita de otros nodos durante los trayectos. Para ello se estudia la posibilidad de inserción de otros nodos antes de la visita de los *R* nodos previamente asignados a las rutas. En este sentido se presenta la segunda de las reglas del algoritmo que consiste en la **regla de inserción** de nodos. Con respecto a la posible inserción de nodos en el camino, es importante destacar varios factores. Se distingue en el algoritmo dos tipos de inserciones: por un lado la **inserción simple** de un nodo intermedio a modo de escala, o bien la **inserción doble** de dos nodos. Estas dos variantes de la regla de inserción persiguen la inclusión de nuevos clientes entre el nodo origen, que en este caso es el depósito, y los nodos destino. Para ello es necesario establecer el valor del tiempo de espera que habría que realizar antes del servicio de cada uno de los nodos semilla, para detectar si existe el tiempo suficiente como para realizar una o dos visitas intermedias. Estas inserciones de nuevos clientes se realizarán finalmente para aquellos nodos intermedios cuya visita implique el menor distanciamiento posible de la línea recta entre el nodo origen y el nodo destino, de manera que se minimiza el desplazamiento adicional que habría que realizar para saturar las esperas ante los nodos semilla con nuevas visitas. La selección de estos nodos intermedios ha de respetar en todo momento las restricciones que imponen las ventanas de tiempo de los mismos, así como las restricciones de carga del camión, y por último las restricciones impuestas por el horizonte temporal, correspondiente al momento de cierre del depósito. En la figura 3.2

se representan todas las posibilidades de inserción entre el depósito central y el nodo 33, según la holgura temporal resultante al considerar el momento de cierre del nodo 33, y que excluye aquellos nodos cuya visita implica llegar fuera de plazo al nodo 33. Estas posibilidades se encontrarán siempre en el área marcada por la elipse dibujada en color azul, indicativa de la holgura temporal existente antes del cierre del nodo 33, aunque solamente será posible la visita de aquellos nodos que se encuentren abiertos en el momento en el que el camión los visitaría, correspondiéndose con los nodos dibujados en rojo.

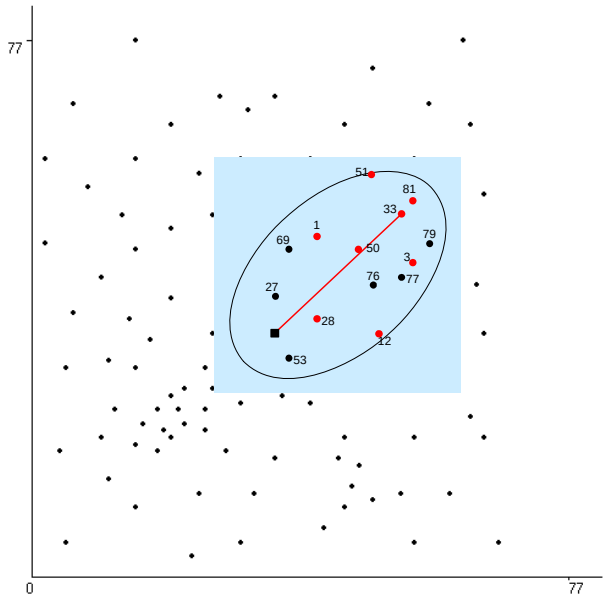


Figura 3.2 Área de inserción establecida según la holgura temporal existente

Igualmente se ha introducido una restricción adicional sobre los desplazamientos que han de realizarse para visitar los nodos intermedios, y que limita la distancia incremental resultante para realizar esas visitas intermedias. Los desplazamientos geográficos permitidos para realizar estas inserciones vienen determinados por sendos parámetros β y γ seleccionados por el usuario y que hacen referencia a la distancia incremental permitida sobre la distancia entre el nodo origen y el nodo destino, de manera que para la inserción simple solamente se contemplarán aquellos nodos intermedios que impliquen una distancia adicional menor o igual a β veces la distancia original entre el nodo origen y el nodo destino. Por ejemplo, si existe una distancia entre el nodo *0* y el nodo en consideración equivalente a *10* unidades, y el parámetro delimitador β es de *2*, entonces el desplazamiento total resultante de la inserción de un nuevo nodo entre los dos en consideración, sería de 2×10 unidades, esto es de *20* unidades. Es decir que solamente se permitiría la visita de aquellos nodos intermedios que provocasen un desplazamiento total entre el nodo *0* y el nodo en consideración menor o igual que *20* unidades. En la figura 3.3 se recogen tres áreas de inserción alternativas. En primer lugar, la elipse mayor indica todas las posibles inserciones de nodos permitidas por la holgura temporal existente en el propio nodo 33. La elipse intermedia se correspondería con la zona de inserción marcada

por un $\beta=2$,de manera que se restringiría el área original a una zona menor, evitando nodos que supondrían amplios desplazamientos. Por último se indica una zona de inserción menor representativa de un $\beta=1.10$, de forma que todavía se limitan más las posibles inserciones de nodos intermedios. De forma análoga podríamos establecer que si el valor de β fuese *1*, entonces solamente se permitirían inserciones de nodos que estuviesen sobre la línea recta.

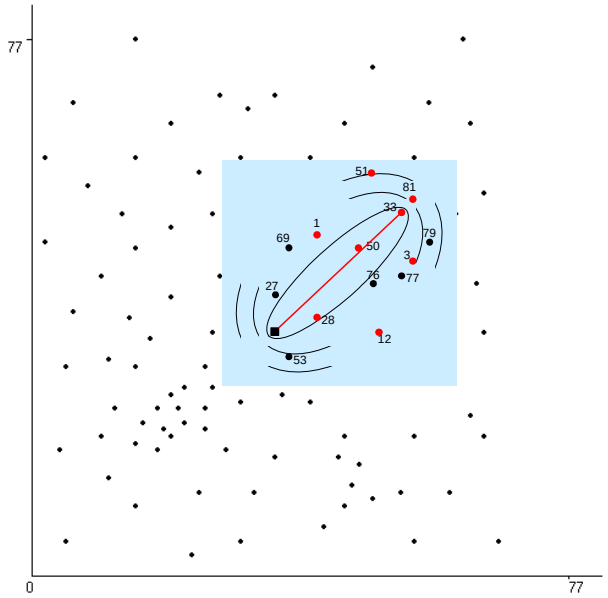


Figura 3.3 Delimitación de la zona de inserción simple a través del parámetro β

En cuanto al parámetro γ es un indicativo penalizador de desplazamientos para el caso de las inserciones dobles, de manera que el desplazamiento máximo autorizado para insertar conjuntamente dos nodos intermedios sería de $\gamma \times \beta$ veces la distancia original entre el nodo origen y el nodo destino.

Una vez que se selecciona el valor de β, entonces se contemplan las diferentes posibilidades de insertar los nodos encontrados en el camino a modo de visitas intermedias entre el nodo origen y el nodo destino. De esta manera se consigue saturar los tiempos muertos o esperas que de otra manera resultarían en las visitas de los nodos inicialmente seleccionados como nodos semilla. En el ejemplo anterior se contemplan las posibilidades de inserción de nodos intermedios entre el depósito central y el nodo 33, para el problema R103, y para el caso de un $\beta=1,10$. Los nodos que podrían visitarse en el camino son los indicados en color rojo en la figura 3.4. Esta delimitación artificial de las distancias incrementales permitidas fomenta la generación de rutas con inserciones muy buenas, a riesgo de computar esperas elevadas y de empeorar la solución final siendo necesario un mayor número de vehículos para realizar todas las visitas.

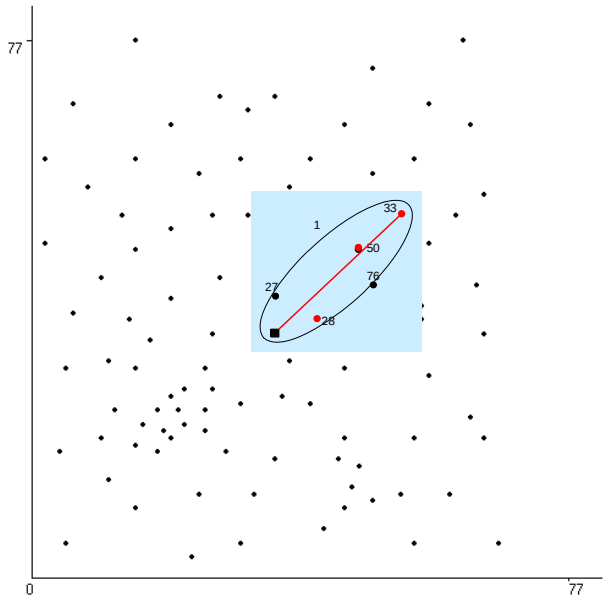


Figura 3.4 Selección de posibles nodos candidatos a ser insertados para un $\beta=1,10$, entre el nodo central y el nodo 33.

Una vez establecidas las diferentes posibilidades de inserción simple se recogen en una lista, entonces se estudia la posibilidad de realizar una inserción doble. Ésta solamente va a ocurrir cuando existan dos ó más nodos en la lista de candidatos a ser inserciones simples. De esta manera se analizan todas las posibles combinaciones de los nodos incluidos en la lista de inserciones simples tomados dos a dos, de manera que se seleccionará el par de nodos que menor desplazamiento provoque con respecto a la distancia entre el nodo origen y el nodo destino. Para el ejemplo anterior se contemplan únicamente dos posibilidades, representadas en las figuras 3.5 y 3.6.

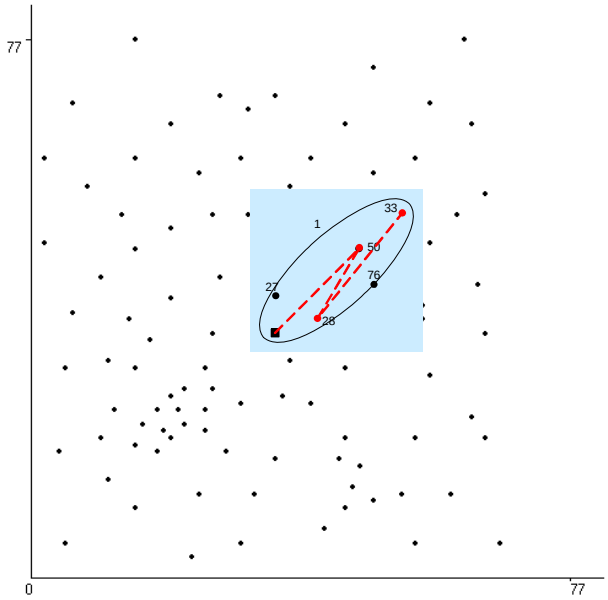


Figura 3.5 Primera posibilidad de combinación para inserción doble

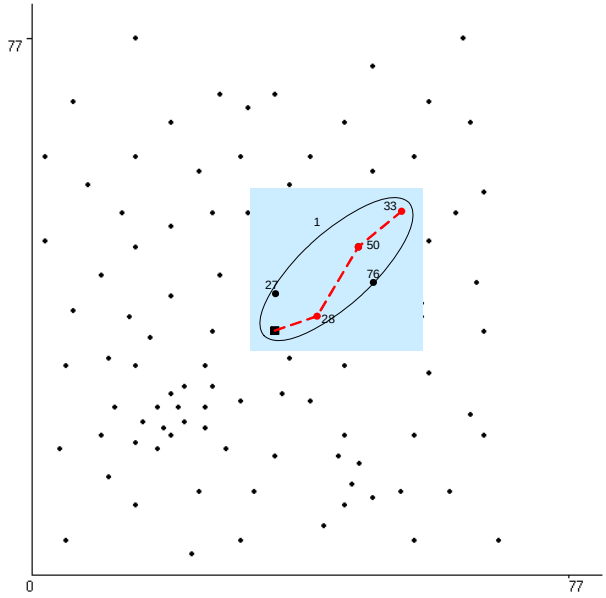


Figura 3.6 Segunda posibilidad de combinación para inserción doble

Una vez seleccionado la secuencia de nodos a ser insertados entre el nodo origen y el nodo destino, que en el ejemplo anterior se corresponde con la segunda alternativa, dado que es la que minimiza el desplazamiento adicional sobre la distancia original, entonces

se procede a la comprobación de que este desplazamiento adicional no supera la delimitación artificial impuesta por la combinación de los parámetros β y γ. La necesidad del parámetro γ es bastante intuitiva si contemplamos dobles inserciones. Es necesario ampliar el área de influencia de forma artificial, ya que generalmente una inserción doble supone un desplazamiento mayor que las inserciones simples, y resultando aún así en una buena estrategia. Por ello es necesario incrementar artificialmente el área de influencia para el caso de las inserciones dobles ya que sino se rechazarían la mayor parte de las inserciones dobles, tal y como ocurriría en para el ejemplo anterior, y como se representa en la figura 3.7. En el caso de la combinación de los nodos *28* y *50* como posibles secuencia de visitas intermedias, el vehículo recorrería una distancia total que sobrepasaría la delimitación de distancia incremental permitida denotada por β y que se corresponde con el área azul, de manera que la suma de las distancias se correspondería con la línea de puntos en color negro, y caería fuera de la zona de influencia, por lo que se rechazaría esa combinación, aún siendo una buena opción.

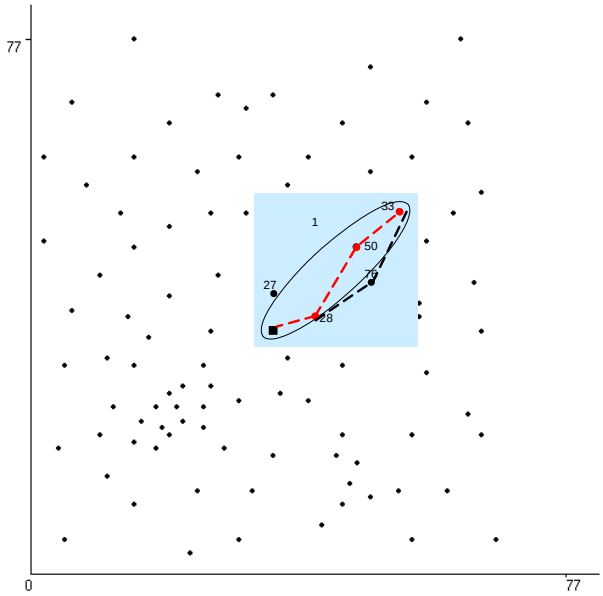


Figura 3.7 Evaluación de la restricción sobre la distancia incremental recorrida por el vehículo.

Por ello para que el modelo no rechace estas buenas combinaciones se establece la necesidad de incrementar esta área de influencia a través del parámetro multiplicador γ, tal y como se recoge en la figura 3.8, para un valor de $\gamma=2$.

De esta manera la selección de la secuencia de nodos cumpliría todas las restricciones impuestas por el modelo, así como las restricciones propias de la temporalidad existente para las ventanas de tiempo de todos los nodos, las restricciones de carga de los vehículos y las restricciones sobre un eventual regreso al nodo central.

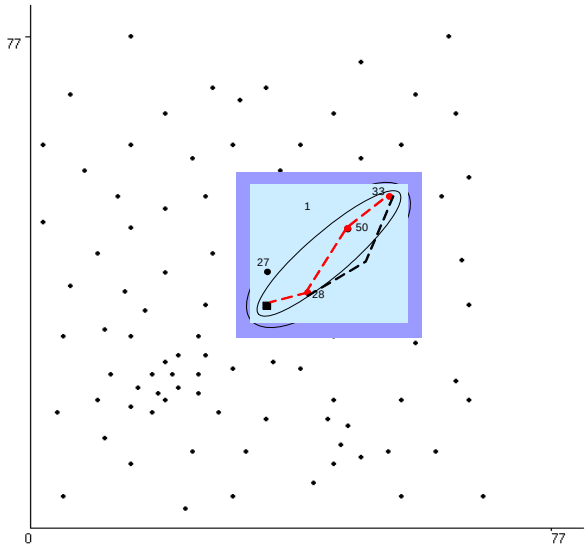


Figura 3.8 Ampliación de la zona de inserción para las inserciones dobles a través del parámetro γ

Así se procedería con todos los nodos semilla. En este sentido el hecho de comenzar por los nodos con menores holguras, implica que la selección de nodos a ser insertados es mucho mejor, ya que se encontrarán en puntos muy cercanos a la línea recta entre el nodo central y el nodo semilla en cuestión. El gráfico resultante de estas inserciones simples y dobles para esta primera etapa de planificación se recoge en la figura 3.9

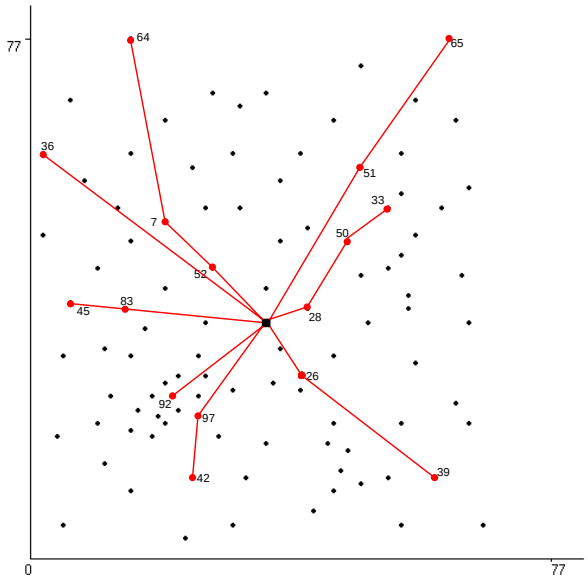


Figura 3.9 Resultado de los primeros nodos de las *R* rutas iniciales, después de aplicar las reglas de asignación, inserción simple, e inserción doble.

Una vez realizado el estudio para estas primeras asignaciones, tiene lugar la segunda y posteriores fases del algoritmo, consistentes en la asignación de los siguientes *R* nodos y sucesivos. Para ello se seleccionan los siguientes *R* nodos en función de los menores tiempos de cierre, equivalentes a los nodos que requieren una atención más inmediata. Una vez seleccionados, se estudian cada uno de ellos detalladamente para su asignación a cada una de las rutas existentes en ese momento. La asignación se hace en todo momento respetando las restricciones temporales impuestas por los tiempos acumulados de cada vehículo, así como los correspondientes tiempos estimados de llegada a los nodos en consideración.

Para realizar esta asignación se seleccionan jerárquicamente los nodos cuyo momento de cierre es más temprano, de forma que se procede a añadirlo a la ruta a la que esté más cercano, siempre que se respeten todas las restricciones. En la figura 3.10 se muestran los siguientes nodos a ser atendidos, así como su asignación a las diferentes rutas existentes.

En el caso de que no fuera posible la asignación de un nodo a una ruta, entonces se procede a la generación de una nueva ruta, como ocurre con el nodo *27* del ejemplo.

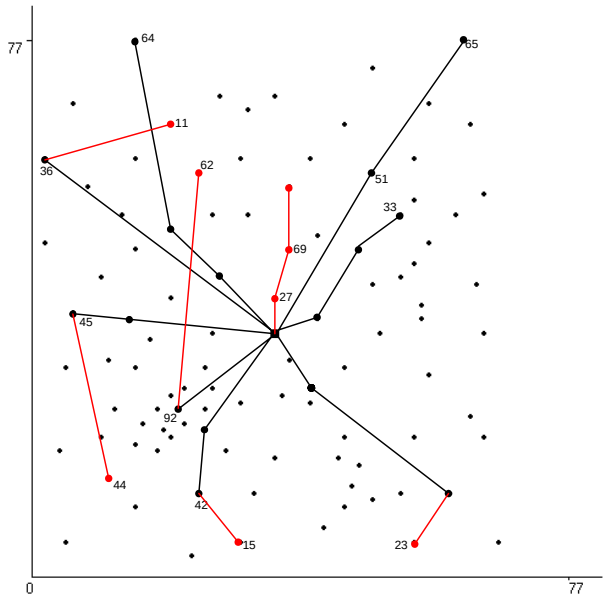


Figura 3.10 Selección y asignación de los siguiente *R* nodos críticos, según sus momentos de cierre.

En la figura 3.10 puede llamar la atención la asignación de los nodos 44 y 92 a sendas rutas, ya que aparentemente no se trata de los vehículos a los que están más cercanos. Sin embargo esto no es así, ya que es necesario tener en cuenta el orden jerárquico de asignación de los vehículos a las rutas, y que en ocasiones implica que un vehículo que está cercano a un nodo determinado se tenga que desplazar de la zona para visitar otro nodo cuyo momento de cierre es anterior, de manera que se pierde la posibilidad de visitar aquellos nodos cercanos, tal y como ocurre con los nodos 44 y 92. Este es uno de

los posibles inconvenientes del algoritmo tal y como se discute en el capítulo 5, aportando posibles soluciones para su corrección.

Una vez asignado un nodo a una ruta, se procede con el estudio de las posibilidades de inserción de nuevos nodos en el camino, tanto individualmente como inserciones dobles. Después de añadir los nodos al vehículo correspondiente, se vuelve a recalcular la lista *R* de nodos críticos. Es importante destacar que la lista se recalcula cada vez que se amplía una ruta con nuevas adiciones, dado que si bien los tiempos de cierre de los nodos de la lista no cambian, lo que sí cambian son las distancias entre estos nodos y las posiciones de los vehículos que se hayan desplazado.

De esta manera se van completando las rutas, hasta que finalmente no quedan nodos. Cuando esto ocurre se finalizan todas las rutas haciendo regresar los vehículos al depósito. El algoritmo procede respetando en todo momento las restricciones temporales de llegadas y salidas a los nodos, al igual que al depósito central, así como las restricciones derivadas de las cargas máximas admitidas por cada camión. Si bien intuitivamente es bastante simple, la complejidad se aborda con la modelización e implementación de todas las reglas de decisión, así como de las propias restricciones del problema.

1.2 Comparativa con otros métodos de construcción

En el capítulo 2 se realizaba una revisión completa de la literatura en la línea de investigación operativa aplicada a la planificación de rutas de transporte. Para ello se presentaba una evolución histórica de las diferentes metodologías y líneas de investigación desarrolladas en el seno de la investigación operativa en los últimos 40 años, incluyendo los problemas del TSP, el VRP y más concretamente el VRPTW. Dentro de esta revisión bibliográfica se enmarcaban los métodos heurísticos de construcción de rutas para el problema del VRPTW, que al mismo tiempo es el ámbito donde se integra el planteamiento del nuevo método heurístico recogido en este trabajo. Por ello en este apartado se establece a modo de comparación un análisis más detallado de algunos de los métodos recogidos anteriormente, buscando en todo momento las similitudes y diferencias con el modelo aquí recogido.

Si bien en el capítulo 2 se presentan métodos anteriores a 1987, es en este año cuando Solomon establece el verdadero punto de partida para los planteamientos heurísticos para la resolución del VRPTW. Para ello aporta un conjunto de problemas estandarizados que sirven a partir de ese momento como punto de partida para todos los modelos desarrollados en la línea de investigación orientada al VRPTW. En ese sentido sirve como plataforma de comparación de la bondad de los métodos desarrollados por todos los autores, constituyendo una verdadera batería de soluciones normalizadas. Por ello, y con el ánimo de contrastar la eficacia y eficiencia del método recogido en este trabajo, solamente se comentan los algoritmos posteriores a esa fecha.

Al mismo tiempo que Solomon (1987) aporta sus conocidos problemas tipo, también realiza una recopilación de los trabajos elaborados hasta esa fecha con la intención de

sintetizar todos los avances en tres nuevos métodos, que ya se comentaban en el capítulo 2, y que se retoman ahora para su comparación con el método propuesto en este trabajo.

El primero de estos métodos consiste en una extensión del conocido método de los ahorros de Clarke y Wright (1964) en el que comienza con una solución en la que cada cliente constituye una única ruta servida por vehículos individuales y a partir de esta solución inicial los clientes se integran en rutas más grandes a través del cálculo de los ahorros generados por estas posibles combinaciones y que equivalen a $S_{ij} = d_{i0} + d_{0j} - d_{ij}$.

En este sentido, el algoritmo que presenta Solomon establece el cálculo de los ahorros para la construcción de rutas más grandes, pero respetando en todo momento las restricciones temporales impuestas por el modelo. Se trata de un algoritmo miope, dado que actúa impulsado por meros cálculos matemáticos. Se trata de unir las mejores combinaciones hasta que ya no es posible conseguir ningún ahorro. Generalmente las soluciones no son demasiado buenas.

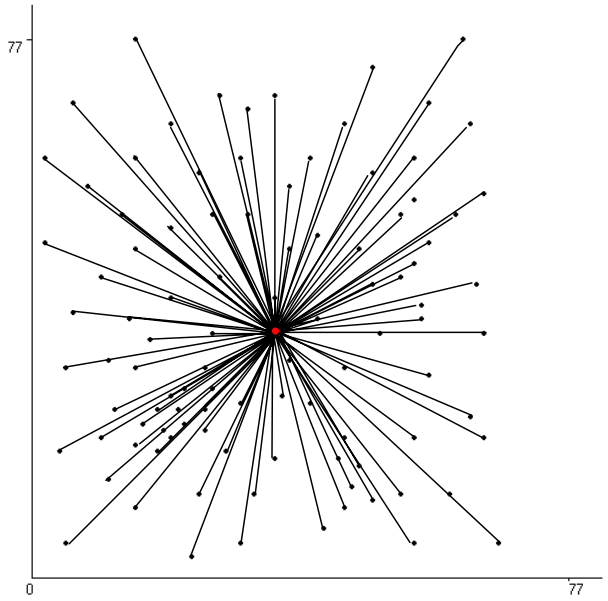


Figura 3.11 Solución de partida para el método *I1* de Solomon (1987)

El segundo método que propone Solomon consiste en un método de adición del cliente más cercano a la ruta, pero en este caso contemplando el criterio temporal. Los pasos que sigue el algoritmo son muy intuitivos. Cada ruta comienza con la selección del cliente que esté más cercano al depósito central. Posteriormente en cada iteración del algoritmo se seleccionan los clientes más cercanos a los últimos clientes añadidos a las rutas, añadiendo estos a las mismas. Una nueva ruta se genera cuando no es posible incluir a un cliente seleccionado en ninguna posición factible. Al igual que en el caso anterior, el

concepto de cercanía incluye tanto el criterio espacial como el criterio temporal. Este segundo método es muy similar al algoritmo presentado en este trabajo.

Por último, el tercer y más exitoso método heurístico propuesto por Solomon es el denominado *I1*. Los pasos del mismo comienzan con la selección de un cliente semilla a partir del cual se añaden el resto de los clientes de la ruta hasta que ésta agota su horizonte temporal, o bien se cubre la capacidad del vehículo. Si quedan clientes sin asignar, entonces se repiten los pasos descritos, hasta que se agotan los clientes. Los clientes semilla seleccionados son aquellos que, o bien son los más distantes del depósito central, o bien los que proporcionan un momento de apertura más temprano para realizar el servicio.

Este tercer algoritmo de Solomon se asemeja al presentado en este trabajo en que ambos contemplan la selección de clientes semilla para realizar la construcción de las rutas. Si bien en el caso de Solomon el proceso de construcción de las rutas es secuencial, mientras que en el caso del presente trabajo las rutas se construyen de forma paralela. Igualmente Solomon selecciona a los clientes semilla bien en función de la lejanía desde el depósito o bien los de apertura más temprana. En el caso del presente trabajo la selección se hace enteramente atendiendo a los criterios de temporalidad, y más en concreto en función de la holgura resultante entre el momento de cierre de cada nodo, y el tiempo necesario hasta llegar a él. El criterio del momento de apertura no parece adecuado teniendo en cuenta las diferentes amplitudes de los nodos, y que puede provocar la visita de nodos cuya apertura es muy temprana, pero que después permanecen abiertos durante mucho tiempo, siendo en ese sentido nodos muy flexibles. En este caso es preferible comenzar por aquellos nodos menos flexibles, y delimitan la planificación, dejando los nodos más flexibles como nodos *"comodín"* para posteriores movimientos de las rutas.

Solomon también contempla la posible inserción de clientes entre cada par de nodos *i* y *j*. Una vez que se construyen las rutas para los nodos semilla, entonces el algoritmo de Solomon empieza a considerar posibles inserciones de los nodos no asignados entre las rutas ya existentes.

Después de iniciar una ruta con uno de estos clientes semilla, el algoritmo de Solomon utiliza dos criterios de inserción, $c_1(i,u,j)$ y $c_2(i,u,j)$ para seleccionar la inserción del cliente *u* entre los clientes adyacentes *i* y *j*. Sea $(i_0,i_1,i_2,\ldots,i_m)$ la ruta en consideración donde i_0 e i_m representan al depósito. Para cada cliente no asignado *u*, se calcula el mejor coste de inserción en una ruta predeterminada de la siguiente manera:

$$c_1(i(u),u,j(u)) = \underset{\rho=1,\ldots,m}{opt}\ c_1(i_{\rho-1},u,i_\rho)$$

A continuación, el mejor cliente u^* para ser insertado en la ruta es aquel para el que

$$c_2(i(u^*),u^*,j(u^*)) = \underset{u}{opt}\, c_2(i(u),u,j(u))$$

donde u es un cliente no asignado a ninguna ruta, y factible con las restricciones, e i y j son los clientes entre los que se insertaría.

Entonces el cliente u^* se inserta en la ruta entre los clientes $i(u^*)$ y $j(u^*)$. Cuando no hay más clientes para ser insertados en las rutas existentes, entonces el algoritmo genera una nueva ruta, a no ser que ya se hayan insertado todos.

Para el cálculo de $c_1(i,u,j)$ se tiene en cuenta lo siguiente

$$c_1(i,u,j) = \alpha_1 c_{11}(i,u,j) + \alpha_2 c_{12}(i,u,j)$$

en donde

$$\alpha_1 + \alpha_2 = 1, \quad \alpha_1 \geq 0, \quad \alpha_2 \geq 0$$

$$c_{11}(i,u,j) = d_{iu} + d_{uj} - \mu d_{ij}, \quad \mu \geq 0$$

$$c_{12}(i,u,j) = b_{ju} - b_j$$

siendo d_{iu}, d_{uj}, y d_{ij} son las distancias entre los clientes i y u, u y j, e i y j respectivamente. El parámetro μ controla los ahorros en la distancia y b_{ju} denota el nuevo momento de llegada al cliente j, suponiendo que el cliente u se haya insertado en la ruta mientras que b_j era el momento de llegada antes de la inserción.

Es realmente en estos dos valores c_{11} y c_{12} donde se aprecian los criterios que utiliza Solomon para establecer estas inserciones. En el primero describe un criterio propio del método de los ahorros, dado que computa un valor equivalente al incremento en el desplazamiento provocado por esa inserción. En este sentido el planteamiento de estas inserciones se puede comparar con el mecanismo de estudio propuesto en este trabajo, que si bien no computa directamente este incremento en el desplazamiento, sí se establece una medición del incremento a través de la reducción de la holgura resultante entre los nodos i y j, también denominada *distancia de inserción* por Dullaert (2000a). En este sentido el algoritmo presentado en este trabajo contempla inicialmente una holgura entre el nodo origen y el nodo destino. Esta holgura se corresponde con el tiempo disponible para realizar posibles escalas intermedias, correspondientes a inserciones de nodos entre el origen y el destino. Por ello se contemplan las nuevas holguras resultantes para las posibles inserciones de nodos intermedios, resultando esta nueva holgura en una medida del desplazamiento adicional provocado.

En este trabajo se mejoran los planteamientos de Solomon en que no solamente es posible insertar nodos individualmente entre cada par de nodos (i,j), sino que también son posibles las consideraciones de pares de nodos simultáneamente. Igualmente el algoritmo presentado en este trabajo contempla una posible limitación del conjunto de posibles nodos a ser insertados, no solamente por sus ventanas de tiempo, sino que en el presente

trabajo también se integra un criterio de desplazamiento geográfico máximo autorizado, de manera que todavía se restringe más el área de posibles nodos a ser insertados, tal y como la define Dullaert en sus trabajos, y que se analiza con más detalle en las próximas líneas.

En el criterio c_{12} Solomon indica el retraso en la llegada al nodo destino provocado por la visita del nodo intermedio. Dicho de otra manera, es la reducción de la holgura resultante por el tiempo de cierre del nodo en cuestión y el tiempo necesario en llegar hasta él. En ese sentido se corresponde en este trabajo con el valor obtenido por la nueva holgura temporal resultante tras haber realizado la inserción de los nodos intermedios, indicativa del tiempo consumido en realizar esa visita intermedia, y que lógicamente tiene mucho que ver con la cercanía o lejanía del nodo finalmente insertado. En ese sentido parece redundante el establecer estos dos criterios, dado que ambos hacen referencia a la misma información.

En resumen, Solomon establece una medida del desplazamiento geográfico provocado por el nodo insertado, y en segundo lugar una medida del tiempo consumido por realizar ese desplazamiento. En este sentido estas dos medidas se refunden en una sola medida en el algoritmo presentado en este trabajo.

Si bien, la dicotomía presentada por Solomon en cuanto a las dos medidas, sí se contempla en la segunda fase del algoritmo expuesto aquí, dado que a partir de la segunda vuelta si es necesario combinar ambas medidas, dado que existen múltiples rutas.

Solomon sin embargo solamente establece este criterio para una elección secuencial de las rutas. Es decir, que las rutas se construyen una tras otra, mientras que en este algoritmo la construcción es paralela, de forma que se aprovechan mejor las oportunidades de inserción que en el caso de una construcción secuencial. Si bien la construcción secuencial tiene la ventaja de que las rutas se van saturando en cuanto a todas las posibles visitas que se pueden realizar, de manera que se reduce el número de vehículos necesarios a costa de incrementarse las distancias recorridas por inserciones en rutas anteriores de nodos que serían buenas opciones para rutas posteriores.

Una vez seleccionado el nodo u a ser insertado, entonces se calcula un segundo criterio denotado por c_2. Este segundo criterio hace referencia a la decisión final de inserción entre los nodos i y j, o bien la creación de una nueva ruta que visite el nodo u directamente desde *0.* El criterio de inserción c_2 se calcula de la siguiente manera:

$$c_2(i,u,j) = \lambda d_{ou} - c_1(i,u,j), \quad \lambda \geq 0$$

donde el parámetro λ se utiliza para ponderar la importancia de mantener el nodo como una ruta separada, frente a la opción de realizar definitivamente la inserción. Se trata de un parámetro de control, que no se ha incluido en el algoritmo aquí presentado, pero que se plantea su integración en el capítulo 5 de Conclusiones y Futuras Líneas de Investigación.

Por último, Solomon propone otros dos métodos de inserción denominados *I2*, en el que las inserciones de clientes se realizan en función a la minimización de la distancia y tiempo total de las rutas, e *I3*, que tiene en cuenta la urgencia de servicio de los clientes.

Dullaert (2000a y 2000b) establece que el criterio de inserción de Solomon $c_{12}(i,u,j)$ subestima el tiempo adicional que supone la inserción del cliente *u* entre el depósito y el primer cliente en las rutas en construcción. Esto puede provocar que se seleccionen subóptimos en las inserciones de estos clientes, por lo que una ruta con un número reducido de clientes puede tener un programa de servicios muy disperso en el tiempo. Dullaert mejora ligeramente los criterios propuestos por Solomon para el caso de rutas muy cortas, lo más interesante de sus trabajos son las reflexiones sobre las posibles inserciones entre cada dos nodos adyacentes. En este sentido analiza alguna de las características que también se encuentran en el algoritmo presentado en este trabajo. Este autor contempla el problema denominado *LTL Routing, (Less Than Truckload Routing),* y que hace referencia a la planificación de rutas en el caso donde no existen restricciones de carga de vehículos dado que se trata de rutas cortas, donde nunca se llega a saturar la carga de los mismos. En ese sentido el autor establece algunas definiciones que merece la pena comentar:

- ***Flexibilidad de Programación***. Para un cliente *i* con un tiempo de servicio ts_i y una ventana temporal (ta_i, tc_i), la flexibilidad de programación se determina por la diferencia entre $tc_i - ta_i$. Es decir que los nodos más flexibles son aquellos que proporcionan mayores ventanas de tiempo, medidas por su amplitud. Por ello es preferible la existencia de este tipo de clientes, ante la existencia de clientes con ventanas de tiempo más reducidas, y que dificultan estas inserciones. En los problemas tipo nos encontramos con estos dos casos, al igual que en la diversidad de problemas con iguales flexibilidades para todos los nodos, o bien diferentes flexibilidades para todos los nodos.
- ***Distancia de Inserción***. Para un cliente *i* con un tiempo de servicio ts_i y una ventana de tiempo (ta_i, tc_i), y un cliente *j,* con un tiempo de servicio ts_j y una ventana de tiempo (ta_j, tc_j), entonces es posible insertar cualquier cliente *u* entre ambos nodos cuya distancia de inserción sea $tc_j - (ta_i+ts_i) - ts_u$. Asumiendo que los tiempos de servicio (*ts*) son siempre iguales, entonces la inserción de un cliente *u* entre *i* y *j* ha de ser tal que:

  $$ta_i+ts+t_{iu} \leq tc_j$$

  e igualmente, también se ha de cumplir que:

  $$ta_i+ts+t_{iu}+ts+t_{uj} \leq tc_j$$

  donde t_{iu} indica el tiempo de desplazamiento entre el nodo *u* y el nodo *i*; *ts* indica la duración del servicio en cualquier nodo, asumiendo la hipótesis de servicios iguales en todos los nodos, y t_{uj} es el tiempo de desplazamiento entre *u* y *j*. Estas

dos restricciones son necesarias para el cumplimiento de los objetivos de cualquier VRPTW, y son las mismas que se contemplan en el planteamiento del modelo recogido en este trabajo. Sin embargo la aportación de Dullaert es la representación del área de inserción elipsoidal delimitada por estas restricciones para cada par de nodos *i* y *j*. Si se suman las restricciones anteriores, se obtiene la siguiente expresión:

$$t_{iu}+t_{uj} \leq tc_j - ta_i - 2 \cdot ts$$

En este sentido la expresión $tc_j - ta_i - 2 \cdot ts$ define la zona de inserción para cada par de nodos *(i,j)*, correspondiente a una elipse que recoge todos los posibles puntos del plano cuya inserción respeta las restricciones temporales del problema. Sin embargo cada vez que se realice una inserción entre cada par de nodos de una ruta parcialmente terminada, es necesario comprobar el cumplimiento de todas las ventanas de tiempo de los nodos posteriores. Si bien el cumplimiento de los nodos anteriores no requiere comprobación, puesto que su construcción ya estaba restringida por sus ventanas de tiempo. Sin embargo sí es necesario comprobar este retraso temporal en los siguientes nodos.

- ***Zona de Inserción:*** La distancia de inserción genera una elipse que puede ser dibujada en el plano como la zona de posibles inserciones. En este sentido, en el algoritmo presentado en este trabajo también se contempla la existencia de zonas de inserción elípticas. Sin embargo a diferencia de Solomon y Dullaert, estas zonas vendrían determinadas por la menor de dos elipses, por un lado la elipse indicativa de la restricción temporal para realizar estas inserciones, y en segundo lugar la elipse determinada por los desplazamientos geográficos máximos permitidos.

De la misma manera en el algoritmo presentado en este trabajo, los nodos incluidos en la zona de inserción así determinada se incluyen en una lista denominada *L2*, de nodos candidatos a ser una inserción simple. Para ello se utilizan los criterios de holguras, de forma que es mucho más sencillo que los cálculos presentados por Dullaert.

Este autor también considera la inserción doble de nodos, que lo denomina *inserción de arcos*, y para ello también establece el cálculo de las zonas de inserción correspondientes, a través del cálculo de las correspondientes elipses, en función de las restricciones oportunas. Estas restricciones son las mismas que fundamentan el algoritmo presentado en este trabajo[18].

Este desglose de conceptos los utiliza para mejorar el desarrollo propuesto por Solomon para las inserciones de nodos entre el depósito y el primer nodo de cada ruta, de manera que consigue afinar más las inserciones de estos primeros nodos, ya que el algoritmo de Solomon infravaloraba únicamente estos primeros nodos. Esto bien se podría haber realizado a través de la restricción de esa elipse de forma artificial, tal y como se propone en el trabajo presentado aquí.

[18] Para mayor detalle referirse a los trabajos de Dullaert (2000a y 2000b).

Solomon (1987) también propone una variante del conocido algoritmo de barrido de Gillett y Miller (1974) a través de la descomposición del problema general del VRPTW en dos fases bien diferenciadas. En la primera fase los clientes se asignan a las rutas, tal y como se hace en el algoritmo original de barrido. En este caso el centro de gravedad es calculado de manera que los clientes se particionan de acuerdo con su ángulo polar. En la segunda fase los clientes son asignados a un vehículo utilizando el criterio de inserción del tipo *I1*. Este desarrollo propuesto por Solomon también sirve de base a algunos desarrollos recogidos en la literatura, como Zhu y Lee (1999).

Este planteamiento comienza de nuevo con la creación de estas particiones de los nodos en grupos de nodos cercanos entre sí, de manera que se constituyen pequeños clusters hasta que se saturan las capacidades de los vehículos. Una vez generados estos clusters, (atendiendo siempre a los ángulos polares generados por estos clientes), entonces se programan las visitas de los mismos por cada uno de los vehículos.

El algoritmo presentado por Solomon comienza seleccionando los nodos iniciales de cada ruta. Las rutas se construyen de forma secuencial, de manera que el nodo primero a ser seleccionado es el nodo que proporciona el menor coste C_1 tal y como se recoge a continuación:

$$Coste(C_1) = -\alpha \cdot t_{0i} + \beta \cdot tc_i + \gamma \cdot \left(\left(\frac{|p_i - p_j|}{360} \right) \cdot t_{0i} \right)$$

El nodo seleccionado ha de ser un nodo que esté cercano al depósito (parámetro α), además ha de ser un nodo que cierre pronto (parámetro β) y además ha de ser un nodo cuyo ángulo polar sea muy similar al ángulo polar descrito por el último nodo añadido a la última ruta, y que se denota por el nodo j (parámetro γ). De esta forma se procede a la selección de los nodos semilla para formar las nuevas rutas teniendo en cuenta la fijación de las importancias relativas de cada uno de los parámetros seleccionados por el usuario.

Una vez seleccionado este nodo semilla, se procede a la adicción de nuevos nodos en la ruta inicializada atendiendo siempre a su coste de inserción, que viene determinado por los siguientes valores:

$$Coste(C_i) = D_k + \phi \cdot W_k + \eta \cdot O_k + \kappa \cdot T_k$$

donde D_k es la distancia total recorrida por el vehículo k, W_k es el tiempo total de viaje del vehículo k, O_k es el valor del posible exceso de carga del vehículo k, y T_k es el valor del posible retraso del vehículo k.

Se trata de un método de construcción secuencial de rutas en función del conocido método del barrido, pero en este caso respetando las restricciones temporales propias del modelo VRPTW. Además cabe destacar que en este planteamiento se recoge la posible violación de algunas restricciones, con la finalidad generalmente de que sirvan de base

para generar gran cantidad de soluciones (algunas no factibles), para posteriores aplicaciones de operadores de mejora.

Potvin y Rousseau (1993) presentan una versión paralelizada del método de inserción de Solomon *I1*, donde un conjunto de m rutas se inician simultáneamente. Los autores utilizan el heurístico de inserción de Solomon para determinar el número inicial de rutas y el conjunto inicial de clientes semilla. La selección del cliente siguiente a ser considerado se basa en una medida del rechazo de inserción en todas las rutas, de manera que un gran rechazo se produce cuando existe un gran espacio entre el mejor y el segundo mejor puesto para ser insertado un cliente.

En este método también se presentan similitudes con el algoritmo desarrollado en este trabajo. Se mejora sustancialmente el modelo inicialmente propuesto por Solomon, ya que en este caso sí se contempla la construcción simultánea de rutas, y no de forma secuencial como lo hacía Solomon. En ese sentido gana en similitud con el algoritmo aquí presentado. Recordemos que el algoritmo recogido en este trabajo genera un primer conjunto de rutas, atendiendo a la decisión de un número *R* de nodos semilla establecido por el usuario, y a partir de ese momento las asignaciones de los siguientes nodos a las rutas se realiza en función a la ruta de la que se encuentren más cercanos. Por ello se asemeja al modelo presentado por Potvin y Rousseau, aunque si bien en este caso el criterio se realiza en función de los rechazos presentados por cada una de las rutas.

En el resto de los trabajos recogidos en la literatura sobre métodos de construcción de rutas, nos encontramos generalmente con el estudio de posibles implementaciones diferentes de los métodos originales de Solomon (Foisy y Potvin, 1993), o incluso en los análisis de aplicabilidad de estos modelos a situaciones reales (Balakrishnan, 1993; Ioannou et al., 2001). Si bien, es necesario destacar que los planteamientos establecidos por Solomon son la base para el desarrollo del resto de los modelos recogidos en la revisión literaria, así como de los propios modelos metaheurísticos. Tal y como se comentaba en el capítulo 2, todos los modelos de postoptimización, ya sean de mejora, o metaheurísticos, han de partir de una o varias soluciones factibles o no de partida. Estas soluciones han de ser obtenidas siempre por los procedimientos de construcción de rutas, y en ese sentido, se seleccionan mayoritariamente los modelos de Solomon, preferentemente el modelo *I1*. Podemos afirmar sin riesgo a equivocarnos, que el 80% de los planteamientos posteriores a Solomon, se fundamentan en sus métodos de construcción de rutas para abordar de forma eminentemente matemática la mejora de las soluciones encontradas por estos algoritmos.

1.3 Conclusiones

En este apartado se establece una serie de conclusiones con respecto al modelo presentado en este trabajo, antes de abordar la formulación matemática del mismo.

- En primer lugar se trata de un algoritmo puramente de construcción de rutas, esto es, construye las rutas partiendo del nodo central desde el primer momento de la planificación, y a partir de ahí se atiende a un criterio temporal para ir

completando estas rutas. De esta manera se presenta como un algoritmo de alto valor añadido para comprender las sucesivas etapas de la planificación de rutas de transporte no como ocurre con otros algoritmos donde las reglas aplicadas se fundamentan en decisiones miopes de agrupación de nodos y programación de los mismos. En este sentido, en cada momento de la planificación propuesta existe la garantía de que todos los nodos anteriores a ese momento de la planificación ya han sido asignados, algo que no ocurre con otros algoritmos, como por ejemplo el de barrido de Gillet y Miller, donde los nodos se asignan en función de sus ángulos polares (procedimiento de barrido). Por ello el algoritmo presentado en este trabajo se presenta como altamente útil para su futura aplicación a entornos dinámicos donde nuevos nodos o visitas surgen a medida que se está ejecutando la planificación inicial. Así sería posible integrar estos nuevos nodos en la planificación para proceder a su asignación, sin que esto rompiese las rutas que se están ejecutando en ese momento. Esto no ocurriría en otros algoritmos donde las rutas se completan de forma secuencial, y no sería posible incluir nuevos nodos sin romper las rutas ya generadas.

- En segundo lugar, cabe destacar los escasos trabajos existentes en la literatura sobre procedimientos de construcción de rutas, ya que puramente de construcción solamente nos encontramos con los trabajos relativos a los procedimientos de inserción propuestos por Solomon, y posteriormente desarrollados por otros autores. El trabajo aquí presentado contempla muchos de los principios desarrollados por Solomon, pero implementados en un conjunto de reglas y decisiones propias que intentan mejorar la planificación originalmente propuesta por Solomon en muy breves espacios de tiempo. Se combinan los procedimientos de adición, inserción simple e inserción doble en unas reglas de asignación de cada nodo crítico a las rutas existentes, fundamentado en un criterio de cercanía de un nodo a una ruta.
- Por último, es necesario tener en cuenta que no se trata de un trabajo enfocado a superar las mejores soluciones encontradas hasta la fecha, sino que el objetivo es profundizar en la comprensión y posterior aplicación del problema teórico del VRPTW a situaciones reales, con la finalidad de aportar un modelo que sirva de apoyo a la toma de decisiones. El método propuesto se aplcica a los problemas tipo de la literatura y se compara con algunos de los modelos existentes. En el capítulo 5 se analizan las posibilidades tanto de mejora del presente modelo como de aplicación a situaciones reales en empresas.

8 Modelización matemática del algoritmo

En esta sección se desarrolla la modelización matemática del algoritmo. Para ello se ha seguido un esquema lógico de trabajo que comienza con las primeras tareas realizadas en el algoritmo. Es necesario tener en cuenta que en este trabajo se ha partido de cero para la construcción del método heurístico, lo cual lo diferencia de la mayor parte de los algoritmos presentados en la literatura y que se servían los unos de los otros para establecer combinaciones entre los mismos tratando siempre de superar las mejores soluciones halladas hasta la fecha.

1.4 Notación

En este apartado se recogen las siguientes variables en el planteamiento de cualquier problema VRPTW, y que sirven para desarrollar las reglas de decisión del algoritmo.

N	Número total de clientes
i	Índice de los clientes: $i = 1, 2, \ldots, N$
x_i	Abscisa de la posición del cliente i
y_i	Ordenada de la posición del cliente i
c_{ij}	Coste de viajar del nodo i al nodo j.
t_{ij}	Tiempo de desplazamiento entre el cliente i y el cliente j
ts_i	Tiempo de servicio del cliente i.
V	Número total de vehículos disponibles
k	Índice de los vehículos. $k=1,\ldots,V$
q	Capacidad de los vehículos en consideración
d_i	Demanda del cliente i
ta_i	Momento de apertura del cliente i
tc_i	Momento de cierre del cliente i
ta_0	Momento de apertura del depósito central
tc_0	Momento de cierre del depósito central
R	Número de nodos semilla a considerar inicialmente por el algoritmo.
β	Parámetro penalizador de desplazamientos en inserciones simples
γ	Parámetro multiplicador de zonas de inserción para inserciones dobles
te_{ki}	Tiempo de espera antes de visitar el nodo i por el vehículo k.
LR_k	Lista de nodos que pertenecen a la ruta k. Se trata de un vector de nodos ordenados que indican las sucesivas visitas que realiza el vehículo k.
tr_k	Momento en el que se encuentra el vehículo k. Esta variable será igual a la suma de los tiempos de desplazamiento entre los nodos que visita este vehículo k, más los tiempos de servicio de los nodos de la ruta k, más las esperas que se han de realizar antes de visitar cada uno de los nodos de la ruta k. Los tiempos de desplazamiento de los nodos será la suma de t_{pq} cuando $q>p$
c_k	Carga total del vehículo k. La carga total del vehículo k es la suma de las demandas de los nodos que ha visitado.

CR_k La cabeza de la ruta k se corresponde con el nodo en el que está ubicado el vehículo k .

D_k Distancia total recorrida por el vehículo k. Esta variable es igual a la suma de los desplazamientos entre los nodos que visita este vehículo.

E_k Espera total que realiza el vehículo k.

1.5 Planteamiento del algoritmo heurístico

En los siguientes apartados se detallan las reglas de decisión que subyacen del algoritmo desarrollado en este trabajo para la resolución del VRPTW. En este sentido se han adaptado algunas de las variables descritas en el modelo general para proceder a la implementación de las reglas propias de este modelo.

1.5.1 Cálculo de la matriz de distancias

El algoritmo calcula una matriz de distancias entre todos y cada uno de los nodos, incluyendo tanto los clientes como el depósito central. Las distancias entre cada par de nodos i y j se calcula a través de la fórmula de Pitágoras que proporciona la distancia euclidea entre dos puntos.

$$d_{ij} = \left[\left(x_j - x_i \right)^2 + \left(y_j - y_i \right)^2 \right]^{1/2}$$

De esta manera se calculan todas las distancias entre cada dos nodos del problema resultando en una matriz de $N+1$ filas y $N+1$ columnas.

Esta matriz de distancias, en el caso del VRPTW también es la matriz de tiempos de desplazamiento, ya que se asume que todos los camiones viajan a una velocidad de 1 unidad de espacio por unidad de tiempo. Esta aclaración es de suma importancia a la hora de abordar la aplicabilidad de los resultados a una planificación real en una empresa con este tipo de problemas. Si bien este tema se plantea posteriormente, vale la pena hacer una serie de aclaraciones. En las operaciones realizadas por el algoritmo se podría actuar de dos formas:

La primera consiste en que la tabla de tiempos de desplazamiento se calcule una sola vez de manera que los datos para los tiempos de desplazamiento entre todos los nodos permanezcan como datos en el problema una vez realizados los cálculos al inicio de la programación de las rutas. De esta manera es necesario que el algoritmo almacene en su memoria gran cantidad de información, que perdurará durante toda la programación realizada. Esta opción tiene la ventaja de que los cálculos se realizan una única vez, de manera que cuando se necesite comprobar un tiempo de desplazamiento entre cualquier par de nodos, solamente será necesario acudir a la matriz para chequear estos datos y utilizarlos como información en todas las fases de la programación. El inconveniente es el consumo de recursos que provoca el almacenar constantemente los $(N+1) \times (N+1)$ datos de tiempos de desplazamiento entre todos los nodos, sabiendo a priori que muchas de estas informaciones son redundante, dado que en ocasiones $t_{ij}=t_{ji}$, o bien porque existen

nodos cuyas uniones son imposibles dado que los tiempos de servicio de los mismos son incompatibles.

La segunda opción es no realizar estos cálculos al principio del problema y realizar los mismos únicamente cuando sean necesarios. De este modo el algoritmo realizará solamente los cálculos de aquellos nodos en consideración en cada momento, por lo que el volumen de datos a utilizar por el algoritmo se reduce, a consta de ampliar el número de operaciones matemáticas a ser realizadas durante la programación. De esta manera se liberan recursos de memoria para que el algoritmo realice un mayor número de operaciones.

Las dos opciones se recogen en los modelos matemáticos presentados generalmente en la literatura. A pesar de ello, cuando se analiza la aplicabilidad de los modelos a la resolución de problemas reales, entonces nos encontramos con que necesariamente se ha de contar con una matriz de distancias o tiempos de desplazamiento, dado que permite mucha más flexibilidad a la hora de realizar planificaciones, además de las posibilidades anteriormente comentadas de manipular esta matriz con datos corregidos, alterados, o simplemente modificados por un sistema de inteligencia artificial.

En este trabajo se ha contemplado la opción de calcular los tiempos de desplazamiento una única vez, de manera que se genera una matriz de $(N+1) \times (N+1)$ datos que se mantendrán durante la programación de las rutas en la memoria del programa en consideración. Estos tiempos de desplazamiento se obtienen por la fórmula de Pitágoras, aunque posteriormente se tratan como datos en el problema lo que permitiría al usuario modificar estos datos con datos reales.

Esta opción se ha escogido para facilitar futuras líneas de investigación en las que se contemple la aplicación del programa a situaciones reales en las que los tiempos de desplazamiento no son euclídeos, y por ello es necesario trabajar con datos reales de desplazamientos entre ciudades.

1.5.2 Definición de parámetros

El siguiente paso es la definición de los parámetros a determinar por el usuario y que integran los siguientes:

8.1.1.1 Número de nodos semilla *R*

R: Número de rutas a considerar inicialmente por el algoritmo. En este sentido el usuario puede optar por dos vertientes. La primera consiste en considerar inicialmente el número de rutas reducido, dejando al propio algoritmo la generación o creación de posteriores rutas cuando esto se considere necesario, y la segunda alternativa consiste en considerar un número elevado de rutas iniciales, de manera que el algoritmo contempla desde el momento inicial la existencia de nuevas rutas que proporcionan más flexibilidad a la planificación inicial.

8.1.1.2 Parámetro *β*

Se trata del parámetro que impide grandes desplazamientos. El algoritmo puede considerar en la programación de las rutas movimientos que aparentemente supongan

grandes desplazamientos. Sin embargo, ha de ser el propio usuario quien determine qué desplazamientos son grandes y cuáles no. El parámetro β hace referencia al número de veces de la distancia original entre dos puntos que se permite como desplazamiento total máximo para realizar una visita intermedia. La opción contra el rechazo de posibles inserciones es la espera, por lo que indirectamente este parámetro β es un parámetro penalizador de las esperas.

El parámetro β tiene una importancia clave en el desarrollo del proceso de programación de las rutas. Este parámetro ha de ser siempre mayor o igual que la unidad, dado que cualquier inserción como mínimo ha de estar sobre la línea recta entre el nodo origen y el nodo destino en consideración. A partir de ahí el desplazamiento máximo permitido con respecto a la línea recta entre dos puntos puede ser del 10%, 20%, 200%, etc, según los valores que pueda tomar el parámetro β (*1.1, 1.2, 2*, etc.)

Si el parámetro β toma valores pequeños, el algoritmo optará por no realizar grandes desplazamientos para insertar nodos y realizar escalas antes de los nodos destino, por ello se minimizarán las distancias recorridas. De esta manera el área que marca los posibles nodos a ser insertados en la ruta se disminuye, discriminando nodos que aunque posibles, se consideran como lejanos. En este sentido el algoritmo reserva estos nodos para posteriores inserciones, quizás en los caminos de regreso de los vehículos de manera que quedan pendientes de asignación para posteriores asignaciones mejores.

Los valores reducidos del parámetro β implica no penalizar las esperas dado que implícitamente estamos asumiendo que es preferible no realizar grandes desplazamientos para visitar nodos lejanos en el camino, antes que esperar. Se supone que el coste de la espera es inferior al coste de los grandes desplazamientos. Sin embargo existe el riesgo de que si se proporcionan valores muy reducidos de este parámetro, y por ello proliferen las esperas en los nodos, entonces nos vamos a encontrar que gran parte del tiempo de los camiones se pierde en las puertas de los almacenes, por lo que será necesario un mayor número de camiones para realizar los servicios a todos los nodos. Por ello puede resultar contraproducente. Existe el riesgo de que estos rechazos provoquen que al finalizar la programación de los nodos críticos, hayan quedado sin visitar nodos que están muy dispersos, resultando una última ruta que empeore el resultado final.

La otra opción es otorgar valores elevados al parámetro β, de manera que los nodos candidatos a ser escalas en la visita a un nodo destino sean más numerosos. De esta manera surgen listas más amplias de candidatos para cubrir las posibles esperas que tendrían que realizar los camiones de otro modo, y se admitirían desplazamientos elevados con tal de que se cubriesen esas esperas. Por ello el proporcionar valores elevados al parámetro β implica penalizar estas esperas de manera que se opte por realizar más visitas, aunque estas supongan desplazamientos superiores.
Implícitamente, si se consideran valores elevados para el parámetro β, nos encontramos con una saturación de los tiempos de espera con visitas intermedias, aunque éstas supongan desplazamientos elevados. De esta manera el número de rutas finales del algoritmo también se ve afectado ya que se reducen las necesidades de nuevos vehículos.

8.1.1.3 Parametro *γ*

El algoritmo presentado en este trabajo contempla la posibilidad de realizar inserciones dobles de dos nodos simultáneamente, entre el nodo origen y el nodo destino. En ese sentido la limitación de desplazamientos grandes impuesta por β, resulta insuficiente para el caso de la inserción de dos nodos simultáneamente, teniendo en cuenta que estas inserciones dobles provocan necesariamente desplazamientos superiores, y que no por ello han de ser penalizadas. En este sentido se plantea la posibilidad de incrementar esta zona de inserción impuesta por el parámetro β. Las dos opciones eran, o bien incrementarlo dentro del propio algoritmo a través de un multiplicador fijo, o bien establecer un nuevo parámetro multiplicador con el que se pudiese flexibilizar el algoritmo. En este caso se optó por la segunda opción de manera que se ha incluido un valor γ a ser fijado por el usuario para incrementar el valor de β originalmente dispuesto para el caso de las inserciones sencillas.

1.6 Procedimiento de cálculo de la solución

1.6.1 Inicio del problema

Una vez introducidos todos los datos necesarios para el algoritmo, entonces da comienzo la planificación de las rutas. En este tipo de problemas, generalmente las variables más críticas son las ventanas de tiempo que proporcionan los nodos y que otorgan al planificador muy breves espacios de tiempo en los que se ha de servir el nodo en cuestión. Por ello en un elevado grado, las variables a tener en cuenta en todo momento son las ventanas de tiempo de estos nodos, además de los tiempos de desplazamiento a los mismos y de los tiempos de servicio. En este sentido las distancias, quedan relegadas a un segundo puesto en importancia, dado que las ventanas de tiempo son restricciones que han de ser respetadas en el problema muchas veces a costa de recorrer mayores distancias, en contra de lo que establecen otras variantes del problema, como el VRPSTW (VRP con ventanas de tiempo no rígidas). Por otro lado es conveniente minimizar el número de vehículos a ser utilizados en la planificación, y que generalmente también tiene una relación inversa con la distancia total recorrida.

En primer lugar, se genera la Lista de Nodos no asignados *L0*. Esta lista recoge todos los nodos del problema en cuestión.

En el momento inicial de la computación todos los tiempos de las rutas son equivalentes a *0*, al igual que las cargas de las mismas. Estas variables son meros contadores acumulativos de las inserciones que se van realizando a cada una de las rutas, al igual que las variables de esperas, y distancias totales recorridas. Se generan las $k=1,...,R$ rutas a tener en cuenta desde el momento inicial de la planificación, y a todas ellas se les añade el nodo *0* como punto de partida. Para todas las rutas $k=1,...,R$ se definen las siguientes variables, utilizando la notación presentada en la sección anterior:

$$LR_k=[0]$$
$$CR_k=[0]$$
$$c_k=0$$

$$tr_k=0$$
$$D_k=0$$
$$E_k=0$$

1.6.2 Generación de la lista *L1* de nodos semilla

El primer paso es la generación de la lista inicial de candidatos a ser servidos por los vehículos que recorrerán las *R* primeras rutas diseñadas, esta primera lista la llamamos *L1,* y contendrá exactamente *R* nodos semilla. *L1* será la lista de nodos candidatos a ser servidos inicialmente por el problema. Tal y como se comentaba al principio, los primeros nodos que han de ser servidos derivan de la consideración de sus ventanas de tiempo. Por ello el algoritmo se ha de centrar en el análisis de las ventanas de tiempo de estos nodos, y en especial en el momento de cierre de los mismos. Este dato es crítico porque marca el plazo límite para visitar estos nodos, por lo que nunca se puede violar esta restricción.

En este sentido es necesario explicar varias cuestiones que son de gran importancia. En primer lugar existe la posibilidad de ordenar los nodos bien por momento de cierre, de manera que aquellos que requieren una atención inmediata son los nodos que cierran antes. O bien por su holgura, que sería el tiempo restante que cada nodo provoca al considerar su visita por cualquiera de los vehículos en consideración. En este sentido se ha optado por la segunda alternativa, es decir que se ordenarán los nodos por su holgura y no por su momento de cierre. Las razones son las siguientes:

Si los nodos se ordenan por las holguras, es posible identificar aquellos que han de ser directamente asignados a rutas cuando no es posible realizar paradas o escalas intermedias, ya que las holguras de los mismos son menores que el menor tiempo de servicio, de manera que ni siquiera los nodos que estuviesen sobre la línea recta sería posibles asignarlos porque su servicio impediría llegar a tiempo al nodo en cuestión. En este sentido los tiempos de cierre no permitirían obtener esta información.

Para ilustrar este comportamiento sirva el siguiente ejemplo. En este caso suponemos tiempos de servicio para todos los nodos equivalentes a *10* unidades temporales. Igualmente se recoge en el gráfico las ventanas temporales de los nodos en cuestión, al igual que otros nodos en los que no hay restricciones temporales. También se representan los tiempos de desplazamiento entre los mismos.

Para el caso del nodo *1*, observamos como se produce una asignación directa, dado que su holgura no permite en ningún caso realizar visitas intermedias. Esta información solamente se obtiene a través del cómputo de la holgura, y no del momento de cierre, y que en este caso sería $h_1=58-50=8<10$ por lo que no es posible realizar ninguna parada intermedia, aunque hubiera nodos que se hallan en camino, como es el caso del nodo *5*.

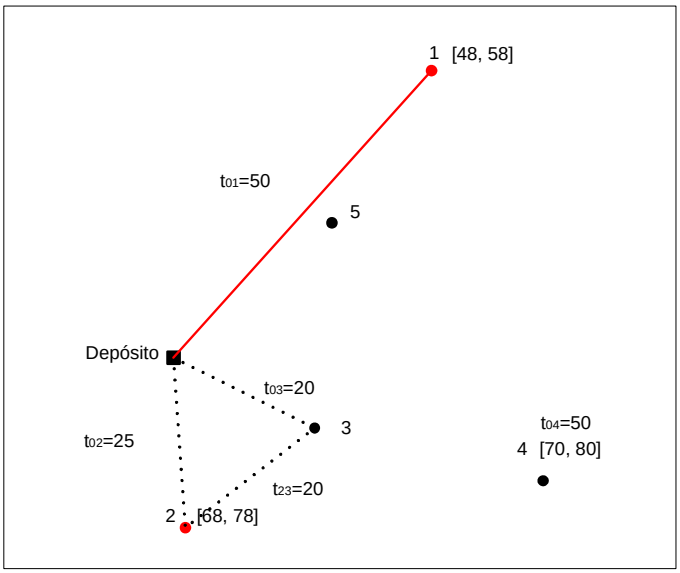


Figura 3.12 Selección de la holgura como valor de referencia para la selección de nodos semilla.

Si se consideran los tiempos de cierre de los nodos, entonces el nodo más crítico sería el nodo *2* antes que otro nodo más lejano (nodo *4*). En este sentido si el nodo *2* permitiese insertar un nodo intermedio a modo de escala, se podría insertar justo el nodo *3* (que suponemos no estaría sujeto a ventanas de tiempo), llegando a tiempo y ampliando la ruta con dos nodos al mismo tiempo, indicado en la línea de puntos.

El siguiente nodo a considerar según tiempo de cierre (nodo *4*), se tendría que añadir a una nueva ruta, dado que ninguna de las rutas existentes llegaría a tiempo. Entonces la opción sería asignarlo directamente a la nueva ruta, realizando una espera antes del servicio del mismo, y sin posibilidad de realizar ninguna escala intermedia, ya que no existirían nodos en camino aceptables.

El resultado es que se habría insertado el nodo *3* antes del nodo *2* para formar una ruta que aparentemente no es tan buena como hubiera resultado la inserción del nodo *3* antes de la visita del nodo *4*. Si se hubiera optado por calcular la holgura, hubiéramos visto que los nodos más críticos (en este caso el nodo *4*) serían aquellos donde las escalas posibles (correspondiente al nodo *3*) deberían siempre de provocar desplazamientos muy reducidos, o lo que es lo mismo, que estuviesen lo más cerca posible de la línea recta, de manera que los nodos "*comodín*" que se van escogiendo suponen muy buenas opciones de inserción.
Dicho de otro modo, esta opción implica que las inserciones intermedias realizadas en el algoritmo no provocan desplazamientos tan distantes como si los nodos se escogiesen atendiendo al momento de cierre de los mismos. Es preferible esperar a que abran los nodos que no incluirlos en desplazamientos grandes, aunque a esta afirmación también contribuye el valor del parámetro β.

Por ello se procede de la siguiente manera para realizar los cálculos para la selección de los nodos semilla. Para cada nodo de la lista *L0* se calcula el tiempo de desplazamiento de *0* a los nodos en cuestión, así como la holgura resultante.

$$h_i=tc_i-t_{0i}$$

A continuación se ordena la lista con todos los nodos de menor a mayor holgura, y se escogen aquellos *R* nodos con holguras menores para formar parte de la lista de nodos semilla *L1*.

1.6.3 Asignación y adición directa a una ruta sin espera

Para aquellos nodos *i* donde la holgura es menor que $min\ ts_i$ (mínimo tiempo de servicio entre todos los nodos del problema) entonces se generan las correspondientes rutas entre el depósito y estos nodos. Para ello se procederá a computar una espera en caso de que la entrada no sea directa en el nodo, caso contrario se entrará directamente a realizar el servicio. Así las rutas generadas inicialmente se actualizan de la siguiente manera con los nodos *i* que implican asignaciones directas:

$$LR_k=[0, i]$$
$$CR_k=[i]$$
$$c_k=d_i$$
$$tr_k=t_{0i} + ts_i + te_{ki}$$
$$D_k=t_{0i}$$
$$E_k=te_{ki}$$

En este caso la espera sería $te_{ki}=[ta_i-(t_{0i})]^+$

Seguidamente se elimina el nodo *i* de la lista *L1*.

Para el resto de los nodos de *L1* se estudia uno a uno comenzando por aquellos con menor holgura, para analizar las posibles inserciones de nodos intermedios entre el depósito central y cada uno de los nodos semilla en consideración.

1.6.4 Generación de la lista *L2* de nodos para la inserción simple

Se selecciona el candidato *i* con menor holgura de entre aquellos que no se han asignado directamente. Después de comprobar su holgura verificamos si ésta es mayor que $min\ ts_i$ y si la holgura supera en *1* ó *2* veces este mínimo tiempo de servicio.
Si su holgura supera el valor de un solo tiempo de servicio, entonces se genera la lista *L2* de posibles nodos intermedios (nodos *m*) para realizar inserciones simples. Los nodos en camino son aquellos cuyo servicio permiten llegar a tiempo al nodo en consideración, y además no suponen un gran desplazamiento (parámetro β), e igualmente no violen las restricciones de carga total del vehículo, así como del tiempo de llegada estimado de regresar al depósito central. Las restricciones aplicables son las siguientes:

Restricción de apertura del nodo *m*

El nodo ha de estar abierto en el momento en que llega el vehículo en cuestión. Por ello se establece la siguiente restricción:

$$ta_m \leq t_{0m}$$

En esta restricción se indica que el momento de apertura de *m* ha de ser anterior a la llegada del vehículo desde *0* hasta *m*.

Restricción de llegada a tiempo al nodo *i* en cuestión.

$$ta_m + ts_m + t_{mi} \leq tc_i$$

Dicho de otra forma, los nodos *m* en consideración como candidatos a ser escalas intermedias son aquellos que permiten llegar al nodo *i* en cuestión antes de que éste cierre. Para ello se contempla un incremento en el tiempo de viaje equivalente al tiempo de desplazarse hasta ese nodo *m* y realizar su servicio, a partir de su apertura, y el consiguiente desplazamiento de *m* a *i*.

Restricción de tiempos de desplazamiento totales

Además es necesario comprobar que la visita de los nodos *m* en cuestión me permite llegar a tiempo al nodo *i* que estoy considerando por eso se establece una restricción temporal consistente en el cómputo de todos los desplazamientos y tiempos de servicio hasta el nodo *i*.

$$t_{0m} + ts_m + t_{mi} \leq tc_i$$

Suponiendo que la visita del nodo *m* en consideración permita llegar antes del momento de cierre, esto provocará un tiempo que se computa como nueva holgura del nodo *i*, de manera que la inecuación anterior resultaría en la siguiente ecuación.

$$t_{0m} + ts_m + t_{mi} + nh_i = tc_i$$

donde nh_i ha de ser mayor o igual que *0*.

Restricción de desplazamiento geográfico máximo autorizado

Es necesario establecer una restricción que evite grandes desplazamientos, y que depende del parámetro β seleccionado. En este sentido solamente se permitirán aquellos desplazamientos para realizar las visitas de los nodos intermedios *m* que supongan un máximo de β veces el desplazamiento original entre el depósito y el nodo *i.*

$$t_{0m}+t_{mi}\leq \beta \cdot t_{0i}$$

1.6.5 Asignación y adición directa a una ruta *k* con espera

Si la lista *L2* está vacía, entonces la opción es esperar y visitar después de la espera el nodo *i* en consideración

Para ello se amplía una ruta *k* comenzando en *0* y visitando el nodo *i* en cuestión, aunque esperando un tiempo equivalente a te_{ki}, y que implica una espera total de la ruta E_k

Se amplía una de las rutas inicializadas al comienzo de la planificación, actualizando todos los valores de la siguiente manera:

$$LR_k=[0, i]$$
$$CR_k=[i]$$
$$c_k=d_i$$
$$tr_k=t_{0i}+ ts_i + te_{ki}$$
$$D_k=t_{0i}$$
$$E_k=te_{ki}$$

En este caso la espera sería $te_{ki}=[ta_i-(t_{0i})]^+$

A continuación se elimina el nodo *i* de la lista *L1*

Si la lista *L2* no está vacía entonces el algoritmo procede a la generación de la lista de posibles inserciones dobles de nodos intermedios, *L3.*

1.6.6 Generación de la lista de inserciones dobles *L3*

Se comprueba si existe posibilidad de realizar una segunda parada antes de realizar el servicio al nodo *i* en estudio. Para ello se genera una segunda lista de dobles paradas denominada *L3*. La lista *L3* está formada por pares de nodos de *L2* que permiten la realización de dos paradas antes del servicio del nodo *i* en consideración. Para ello se seleccionan de *L2* el conjunto n_2 de pares de nodos *m* y *l* tales que:

Restricción de apertura de los nodos *m* y *l*

Cumplen la restricción temporal expuesta para *L2,* o lo que es lo mismo, se trata de nodos que están abiertos para realizar la escala intermedia. Estas restricciones son las siguientes:

$$ta_m \leq t_{0m}$$

$$ta_l \leq t_{om}+ts_m+t_{ml}$$

Restricción de llegada a tiempo al nodo *i* en cuestión

Cumplen la restricción de llegada a tiempo al nodo *i*, en caso de visita de ambas escalas intermedias de los nodos *m* y *l.*

$$t_{0m}+ts_m+ t_{ml}+ts_l+t_{li} \leq tc_i$$

Incluyendo la variable de holgura para deshacer la inecuación, la expresión anterior resultaría en:

$$t_{0m}+ts_m+ t_{ml}+ts_l+t_{li}+ nh'_i = tc_i$$

donde nh'_i ha de ser mayor o igual que *0.*

Restricción de desplazamiento geográfico máximo autorizado.

Es necesario comprobar que no implican un gran desplazamiento, por lo que el desplazamiento total ha de ser menor o igual que γ veces el valor de β veces la distancia original. Matemáticamente:

$$t_{0m}+ t_{ml}+t_{li} \leq \gamma\ \beta\ t_{0i}$$

Una vez computadas todas las nuevas holguras, se procede de la siguiente manera:

Si *L3* está vacía, entonces escogemos el nodo *m* de *L2* que maximiza nh_i, y se añaden a una de las rutas creadas al principio de la siguiente manera:

$LR_k=[0, m, i]$
$CR_k=[i]$
$c_k= d_m + d_i$
$tr_k=t_{0m} + ts_m+t_{mi}+ts_i+te_{ki}$
$D_k= t_{0m} + t_{mi}$
$E_k=te_{ki}$

En este caso $te_{ki}=[ta_i-(t_{0m} + ts_m + t_{mi})]^+$

A continuación se elimina el nodo *i* de la lista *L1* y se elimina el nodo *m* de la lista *L0*

Si *L3* no está vacía entonces se seleccionan los nodos *m* y *l* que maximizan el valor de la nh'_i de manera que se procede a añadir estos dos nodos a una de las rutas originalmente creadas al principio de la planificación de la siguiente manera:

$LR_k=[0, m, l, i]$
$CR_k=[i]$
$c_k= d_m + d_l + d_i$
$tr_k= t_{0m} + ts_m + t_{ml} + ts_l + t_{li}+ts_i+te_{ki}$
$D_k= t_{0m} + t_{ml}+ t_{li}$
$E_k=te_{ki}$

En este caso el $te_{ki}=[ta_i- (t_{0m} + ts_m + t_{ml} + ts_l + t_{li})]^+$

A continuación se elimina el nodo *i* de la lista *L1* y se eliminan los nodos *m* y *l* de la lista *L0*

El hecho de escoger aquel o aquellos nodos que minimicen el desplazamiento total, o lo que es lo mismo, que maximizan la nueva holgura resultante para el nodo *i* en estudio, implica lo mismo que escoger aquel o aquellos nodos que estando entre los dos nodos en consideración, minimizan la desviación sobre la pendiente original, en este sentido se minimiza la desviación generada por los nodos insertados y que serán escalas antes de llegar al nodo destino.

De esta manera se procede a la construcción de las *R* primeras rutas de la solución, con los primeros *R* nodos semilla seleccionados, hasta que se vacía la lista *L1*.

1.6.7 Regeneración de la lista de los *R* siguientes nodos críticos *L1*

Hasta el momento se han construido un total de *R* rutas iniciales y se han actualizado todos los valores para cada una de las mismas y que servirán para esta segunda etapa de planificación. Esta segunda fase se desarrolla de la siguiente manera. La idea consiste en ir seleccionando en cada una de las iteraciones del problema un total de *R* nodos para su asignación a las *R* rutas ya existentes. En este sentido se trata de un algoritmo de construcción paralela dado que las rutas se construyen simultáneamente en contra de los procedimientos de construcción de rutas secuenciales.

Una vez seleccionados los *R* siguientes nodos a ser asignados se buscan aquellas rutas en las que su inserción es factible en términos de temporalidad, y en segundo lugar se tiene en cuenta los criterios de cercanía de cada uno de los nuevos nodos a las cabezas de ruta existentes. Una vez asignados los nodos en consideración se vuelven a contemplar las posibilidades de inserción simple y doble.

A partir de esta segunda etapa se procede a tener en cuenta los tiempos de cierre, y no las holguras como se realizó en la primera fase de la planificación. En este sentido nos encontramos con los nodos que requieren una atención más inmediata en la planificación de las rutas, y que por ello han de ser asignados inmediatamente a las rutas existentes. Estos nodos pasan a conformar la lista *L1*, que en este caso se ordena de forma creciente atendiendo a los momentos de cierre de cada uno de los *R* nodos seleccionados.

1.6.8 Procedimiento de asignación de un nodo *i* a una ruta *k*

Se seleccionan los *R* nodos con momentos de cierre más tempranos de la lista de nodos no asignados *L0*. A continuación estos nodos se ordenan de forma ascendente teniendo en cuenta sus momentos de cierre, de forma que los nodos con momentos de cierre más tempranos son aquellos que primero serán atendidos. En este caso, en contra de lo que ocurría al comienzo de la fase de planificación, la holgura ya no es una medida válida, debido a que para cada uno de los nodos en consideración de la lista *L1* , las holguras son

diferentes según la ruta que se considere. Por ello se opta primero por ordenar los nodos según su requerimiento de inmediatez representado ahora por su momento de cierre, para posteriormente analizarlos en detalle.

Para estos nodos se genera una tabla en la que establecen los cálculos de los tiempos de llegada estimados desde cada una de las cabezas de ruta existentes hasta cada uno de los nodos en consideración. El cómputo del tiempo estimado de llegada del vehículo que está sirviendo la ruta *k* al nodo en consideración *i* equivale al cálculo de $tr_k+t_{CRk\,i}$

Es decir que se computa el tiempo acumulado de la ruta k en cuestión más el tiempo de desplazamiento entre el último nodo de la ruta *k,* denominado *cabeza de ruta k,* hasta el nodo *i* en cuestión.

Este cómputo se realiza para todos y cada uno de los nodos integrados en *L1*, y para todas las rutas existentes en ese momento, de manera que obtenemos una matriz con *L1* filas y *R* columnas donde se disponen todos los valores del problema.

A continuación se comparan estos tiempos estimados de llegada con los momentos de cierre de cada uno de los nodos de la lista *L1* para verificar si la llegada de los vehículos se produce a tiempo para realizar el servicio. De esta forma es posible volver a calcular unas holguras que van a servir para decidir cuáles de los nodos serán asignados a continuación. El cálculo de las holguras para un nodo *i* a ser visitado por la ruta *k* se realiza de la siguiente manera:

$$h_i=tc_i-[tr_k+t_{CRk\,i}]$$

En este sentido solamente se pueden considerar las holguras positivas dado que en caso de resultar negativa implicaría que el vehículo de esa ruta no llegaría a tiempo de realizar el servicio de ese nodo.

Una vez obtenidos estos valores, entonces se selecciona el nodo con cierre más temprano y se estudia en detalle. A este nodo en consideración se denota como nodo *i*, y es el que tiene $min\ tc_i$ para los nodos de la lista *L1*. Entonces se procede al estudio de la holgura de este nodo *i*.

La primera operación que se realiza es la de asignación del nodo *i* a una ruta *k* existente, siempre y cuando la holgura resultante sea positiva. Si esto no ocurre, entonces es que no existe ninguna ruta a la que poder asignar este nodo *i*, por lo que se procedería a la creación de una nueva ruta, tal y como se describía al inicio del algoritmo.

Sin embargo, si existiesen holguras resultantes positivas, entonces se escoge aquella ruta a la que esté más cercano, y que no tiene por qué ser la de menor holgura. Esta consideración es muy importante. La asignación a una ruta determinada se hace en función del criterio de cercanía del nodo en cuestión a esa ruta, y no de la holgura. La holgura es un valor que se utiliza para establecer si el vehículo llega a tiempo o no, y si

llegase a tiempo, entonces sirve para analizar la posible inserción de nodos intermedios, según sea la amplitud de la holgura.

1.6.9 Procedimiento de adición de un nodo *i* a una ruta *k*

Una vez asignado un vehículo a una ruta, entonces se procede al estudio de la holgura. Si esta es inferior al menor valor de los tiempos de servicio ($min\ ts_i$) , entonces no es posible realizar ningún tipo de parada intermedia, por lo que se procedería a la adición directa de este nodo a la cabeza de ruta *k*, de manera que la actualización de los valores de la ruta *k* serían los siguientes[19] siempre y cuando se cumplan las restricciones:

Llegada a tiempo al nodo *i* desde la ruta *k*

$$tr_k + t_{CRk\,i} \leq tc_i$$

Eventual llegada al depósito antes de que éste cierre

$$tr_k + t_{CRk\,i} + ts_i + t_{i0} + te_{ki} \leq tc_0$$

donde $te_{ki}=[ta_i-(tr_k+t_{CRk\,i})]^+$

Cumplimiento con la carga total del vehículo

$$c_k + d_i \leq q$$

Los valores resultantes de la ruta *k* a la que se añade el nodo *i* serían los siguientes:

$$LR'_k=[\,LR_k,\,i]$$
$$CR'_k=[i]$$
$$c'_k=c_k+d_i$$
$$tr'_k=tr_k+t_{CRk\,i}+ts_i+te_{ki}$$
$$D'_k=D_k+t_{CRk\,i}$$
$$E'_k=E_k+te_{ki}$$

En este caso la espera es $te_{ki}=[\,ta_i-(tr_k+t_{CRk\,i})]^+$

1.6.10 Generación de la lista *L2* de escalas intermedias entre el final de la ruta *k* y un nodo *i*

Si la holgura resultante es superior al valor del mínimo tiempo de servicio, entonces se procede al análisis de las posibles inserciones de nodos entre CR_k y el nodo *i* en consideración. Para ello se genera una lista *L2* de posibles inserciones. Esta lista *L2* va a estar compuesta por nodos *m* para los que se cumplen una serie de condiciones, al igual que ocurría al principio, aunque ahora el nodo de partida en vez de ser el nodo *0*, será el

[19] Se ha utilizado la notación con apóstrofe, indicando los nuevos valores de la ruta y que son acumulativos de los valores anteriores.

nodo correspondiente a la *cabeza de ruta k*. Por ello las restricciones correspondientes serán las siguientes para los nodos *m* a ser insertados entre CR_k y el nodo *i:*

Restricción de apertura del nodo *m*

El nodo *m* en consideración ha de estar abierto para cuando llegue el vehículo procedente del nodo correspondiente a la *cabeza de ruta k.*

$$ta_m \leq tr_k + t_{CRk\,m}$$

Restricción de llegada a tiempo al nodo *i* en cuestión

Los nodos *m* han de estar abiertos para realizar su visita antes de que cierre el nodo i.

$$ta_m + ts_m + t_{mi} \leq tc_i$$

Restricción de tiempo de desplazamientos total

En segundo lugar la visita del nodo *m* desde el final de la ruta *k*, ha de permitir la llegada a tiempo al nodo *i*.

$$tr_k + t_{CRk\,m} + ts_m + t_{mi} \leq tc_i$$

Suponiendo que la visita del nodo *m* en consideración permita llegar antes del momento de cierre, esto provocará un tiempo que se computa como nueva holgura del nodo *i*, de manera que la inecuación anterior resultaría en la siguiente ecuación.

$$tr_k + t_{CRk\,m} + ts_m + t_{mi} + nh_i = tc_i$$

donde nh_i ha de ser mayor o igual que *0*.

Restricción de desplazamiento geográfico máximo autorizado

No ha de suponer un gran desplazamiento ya que solamente se permitirán aquellos desplazamientos para realizar las visitas de los nodos intermedios *k* que supongan un máximo de β veces el desplazamiento original entre la cabeza de la ruta *k* y el nodo *i.*

$$t_{CRk\,m} + t_{mi} \leq \beta\, t_{CRk\,i}$$

Eventual llegada estimada de retorno al depósito antes que éste cierre

$$tr_k + t_{CRk\,m} + ts_m + t_{mi} + ts_i + t_{i0} + te_{ki} \leq tc_0$$

siendo $te_{ki} = [ta_i - (tr_k + t_{CRk\,m} + ts_m + t_{mi})]^+$

Cumplimiento con la carga total del vehículo

$$c_k+d_m+d_i \leq q$$

Si la lista *L2* está vacía, entonces la opción es esperar y visitar después de la espera el nodo *i* en consideración. Para ello se actualizan los valores de la ruta *k* en cuestión de la siguiente manera:

$$LR'_k=[\ LR_k,\ i]$$
$$CR'_k=[i]$$
$$c'_k=c_k + d_i$$
$$tr'_k=tr_k + t_{CRk\,i}+ ts_i+ te_{ki}$$
$$D'_k=D_k + t_{CRk\,i}$$
$$E'_k=E_k + te_{ki}$$

En este caso la espera sería $te_{ki}=[ta_i - (tr_k + t_{CRk\,i})]^+$

A continuación se elimina el nodo *i* de la lista *L1*

1.6.11 Generación de la lista *L3* de dobles escalas intermedias entre el final de la ruta *k* y un nodo *i*

Si la lista *L2* no está vacía entonces el algoritmo procede a la generación de la lista de posibles inserciones dobles, *L3* entre la cabeza de la ruta *k* y el nodo *i* en consideración. Para ello selecciona pares de nodos de *L2* para generar la lista *L3* de los nodos a ser insertados simultáneamente.Estos nodos *m* y *l* cumplen las siguientes restricciones:

Restricción de apertura de los nodos *m* y *l*

$$ta_m \leq tr_k + t_{CRk\,m}$$

$$ta_l \leq tr_k + t_{CRk\,m}+ts_m+t_{ml}$$

Restricción de llegada a tiempo al nodo *i* en cuestión

Cumplen la restricción de llegada a tiempo al nodo *i*, en caso de visita de ambas escalas intermedias de los nodos *m* y *l.*

$$tr_k+ t_{CRk\,m}+ts_m+ t_{ml}+ts_l+t_{li} \leq tc_i$$

Incluyendo la variable de holgura para deshacer la inecuación, la expresión anterior resultaría en:

$$tr_k+ t_{CRk\,m} +ts_m+ t_{ml}+ts_l+t_{li}+ nh'_i = tc_i$$

donde nh'_i ha de ser mayor o igual que *0.*

Restricción de desplazamiento geográfico máximo autorizado

Además de las restricciones anteriores, es necesario comprobar que no implican un gran desplazamiento, y en este caso el desplazamiento total será menor o igual que γ veces el valor de β veces la distancia original. Matemáticamente:

$$t_{CRk\,m} + t_{ml} + t_{li} \leq \gamma\ \beta\ t_{CRk\,i}$$

Eventual llegada estimada de retorno al depósito antes que éste cierre

$$tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li} + ts_i + te_{ki} + t_{i\,0} \leq tc_0$$

donde $te_{ki}=[ta_i-(\ tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li})]^+$

Cumplimiento con la carga total del vehículo

$$c_k + d_m + d_l + d_i \leq q$$

Una vez computadas todas las nuevas holguras para todas las posibles combinaciones de nodos *m* y *l*, se procede de la siguiente manera:

Si *L3* está vacía, entonces escogemos el nodo *m* de *L2* que maximiza nh_i, y se añaden a una de las rutas inicializadas al principio de la siguiente manera:

$$LR'_k=[LR_k,\ m,\ i]$$
$$CR'_k=[i]$$
$$c'_k= c_k + d_m + d_i$$
$$tr'_k=tr_k + t_{CRk\,m} + ts_m + t_{mi} + ts_i + te_{ki}$$
$$D'_k= D_k + t_{CRk\,m} + t_{mi}$$
$$E'_k=E_k + te_{ki}$$

En este caso $te_{ki}=[ta_i-(tr_k + t_{CRk\,m} + ts_m + t_{mi})]^+$

A continuación se elimina el nodo *i* de la lista *L1* y se elimina el nodo *m* de la lista *L0.*

Si *L3* no está vacía entonces se seleccionan los nodos *m* y *l* de *L3* que maximizan el valor de la nh'_i de manera que se añaden estos dos nodos a la ruta *k* de la siguiente manera:

$$LR'_k=[LR_k,\ m,\ l,\ i]$$
$$CR'_k=[i]$$
$$c'_k= c_k + d_m + d_l + d_i$$
$$tr'_k=tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li} + ts_i + te_{ki}$$
$$D'_k=D_k + t_{CRk\,m} + t_{ml} + t_{li}$$
$$E'_k=E_k + te_{ki}$$

En este caso el $te_{ki}=[ta_i-(tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li})]^+$

A continuación se elimina el nodo *i* de la lista *L1* y los nodos *m* y *l* de la lista *L0*

De esta manera se procede hasta que todos los nodos han sido asignados, y finalmente se genera la solución final con el detalle de todas las rutas.

1.6.12 Generación de la solución final

Una vez que se han terminado todos los nodos de las listas, entonces se procede a la inclusión del nodo *0* para todas las rutas generadas, de manera que se hace regresar a todos los vehículos al nodo original. Igualmente se actualizan todos los valores de las rutas procediendo a los siguientes cálculos para toda ruta *k*:

$$LR'_k=[LR_k, 0]$$
$$CR'_k=[0]$$
$$c'_k= c_k$$
$$tr'_k=tr_k + t_{CRk\,0}$$
$$D'_k= D_k + t_{CRk0}$$
$$E'_k=E_k$$

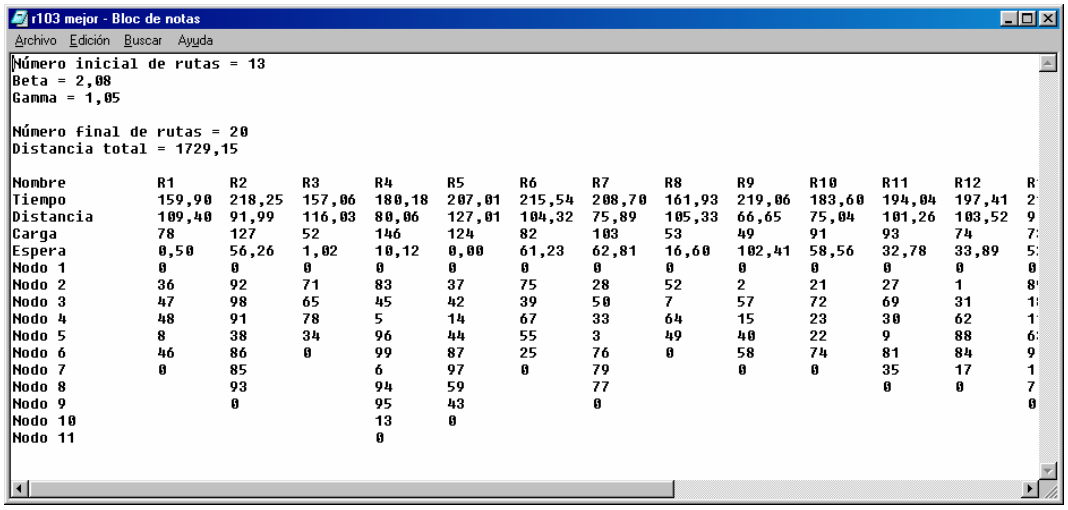
Finalmente el algoritmo genera una tabla con los datos resumen de cada una de las rutas. Esta tabla contiene la solución del problema aportada por el algoritmo, y su diseño obedece al siguiente esquema.

- **Nombre de las rutas**: en la primera fila de la tabla se incluyen los nombres de las rutas creadas.
- **Momento de la Ruta:** Esta variable hace referencia al momento de tiempo en el que se halla la mencionada ruta. Este momento será la suma de los tiempos de desplazamiento que se hayan realizado dentro de la ruta, los tiempos de servicio en los que se haya incurrido, y finalmente las esperas que haya tenido que sufrir el vehículo en cuestión. Esta variable es de suma importancia en el problema del VRPTW, dado que todos los nodos tienen unos momentos determinados de servicio.
- **Carga de la ruta:** Esta variable hace referencia al contador de las cargas que secuencialmente va recogiendo o sirviendo el vehículo en cuestión, y que nunca puede superar la carga máxima del camión.
- **Lista de nodos visitados en la ruta:** La lista de los nodos visitados, se corresponde con el programa de visitas que la ruta en cuestión ha realizado, y que siempre comienza en el nodo *0* y finaliza en el nodo *0*. es importante destacar que el orden de esta lista señala el programa de las visitas, por lo que cualquier alteración en este orden altera todas las variables consideradas anteriormente. Por ello los nuevos nodos en consideración se añadirán siempre al final de esta lista, de manera que el último nodo añadido representará la cabeza de ruta en el momento indicado, que será utilizada siempre como referente de la ruta en

cuestión para detectar los nodos más cercanos a la misma, así como las rutas más cercanas a un nodo.

- **Cabeza de ruta:** Último nodo añadido a la tabla
- **Distancia recorrida por esa ruta**

Finalmente se computarán la suma de las distancias recorridas por todas las rutas, y se representan los resultados en un gráfico.


r103 mejor - Bloc de notas

Archivo Edición Buscar Ayuda

```
Número inicial de rutas = 13
Beta = 2,08
Gamma = 1,05

Número final de rutas = 20
Distancia total = 1729,15

Nombre          R1      R2      R3      R4      R5      R6      R7      R8      R9      R10     R11     R12     R
Tiempo          159,90  218,25  157,06  180,18  207,01  215,54  208,70  161,93  219,06  183,60  194,04  197,41  2
Distancia       109,40  91,99   116,03  80,06   127,01  104,32  75,89   105,33  66,65   75,04   101,26  103,52  9
Carga           78      127     52      146     124     82      103     53      49      91      93      74      7
Espera          0,50    56,26   1,02    10,12   0,00    61,23   62,81   16,60   102,41  58,56   32,78   33,89   5
Nodo 1          0       0       0       0       0       0       0       0       0       0       0       0       0
Nodo 2          36      92      71      83      37      75      28      52      2       21      27      1       8
Nodo 3          47      98      65      45      42      39      50      7       57      72      69      31      1
Nodo 4          48      91      78      5       14      67      33      64      15      23      30      62      1
Nodo 5          8       38      34      96      44      55      3       49      40      22      9       88      6
Nodo 6          46      86      0       99      87      25      76      0       58      74      81      84      9
Nodo 7          0       85              6       97      0       79              0       0       35      17      1
Nodo 8                  93              94      59              77                              0       0       7
Nodo 9                  0               95      43              0                                               0
Nodo 10                                 13      0
Nodo 11                                 0
```

Figura 3.13 Informe de la solución final a través del algoritmo desarrollado

1.7 Consideraciones sobre la utilización de listas

Es necesario realizar algunas consideraciones en la resolución de los problemas tipo a través de este algoritmo:

El problema del VRPTW implica un mayor número de restricciones que el problema clásico del VRP, ya no solamente porque es necesario minimizar la distancia total recorrida, sino también porque es necesario respetar las restricciones de carga impuestas para cada ruta, así como las ventanas de tiempo de cada uno de los nodos, y los tiempos de servicio de los mismos.

En particular tiene especial importancia el hecho de que, si bien el número de nodos es conocido de antemano, así como sus localizaciones, generalmente solamente se puede contemplar un determinado grupo de nodos en cada una de las fases de optimización. Por ello generalmente el número de posibilidades de combinación de los nodos disminuye, en perjuicio del resultado final en cuando a la distancia total recorrida.

En este sentido facilita al usuario determinadas elecciones de nodos provocado por el gran número de restricciones existentes, pero dificulta enormemente la utilización de algoritmos de optimización dado que se incrementan el número de restricciones por lo que es necesario tener un mayor número de cálculos para conseguir un resultado.

En este marco se plantea la utilización de listas para reducir el número de cálculos a realizar por los algoritmos en cuestión, al igual que hacen los planificadores en las empresas. Consecuentemente del total de nodos existentes en el problema se discriminan todos aquellos nodos que no son posibles, escogiendo únicamente los nodos que realmente se pueden asignar en cada una de las fases de ejecución de las rutas.

El planificador requiere en todo momento la atención de los nodos con cierres más tempranos ya que son los clientes que han de ser servidos primero, posponiendo la programación de los que han se ser servidos al finalizar el día. Por ello el planificador reduce el número de nodos en estudio para poder obtener soluciones inmediatas a problemas inminentes.

Generalmente los algoritmos que trabajan con listas realizan el mismo proceso dado que reducen el número de nodos en estudio para eliminar todas aquellas combinaciones que no son posibles porque violan determinadas restricciones.

1.8 Modelo matemático del algoritmo (Resumido)

La notación empleada es la siguiente:

N	Número total de clientes
i	Índice de los clientes: $i = 1, 2, \ldots, N$
x_i	Abscisa de la posición del cliente i
y_i	Ordenada de la posición del cliente i
c_{ij}	Coste de viajar del nodo i al nodo j.
t_{ij}	Tiempo de desplazamiento entre el cliente i y el cliente j
ts_i	Tiempo de servicio del cliente i.
V	Número total de vehículos disponibles
k	Índice de los vehículos $k=1,\ldots,V$
q	Capacidad de los vehículos en consideración
d_i	Demanda del cliente i
ta_i	Momento de apertura del cliente i
tc_i	Momento de cierre del cliente i
ta_0	Momento de apertura del depósito central
tc_0	Momento de cierre del depósito central
R	Número de nodos semilla a considerar inicialmente por el algoritmo.
β	Parámetro penalizador de desplazamientos en inserciones simples
γ	Parámetro multiplicador de zonas de inserción para inserciones dobles
te_{ki}	Tiempo de espera antes de visitar el nodo i por el vehículo k.
LR_k	Lista de nodos que pertenecen a la ruta k. Se trata de un vector de nodos ordenados que indican las sucesivas visitas que realiza el vehículo k.
tr_k	Momento en el que se encuentra el vehículo k. Esta variable será igual a la suma de los tiempos de desplazamiento entre los nodos que visita este vehículo k, más los tiempos de servicio de los nodos de la ruta k, más las esperas que se han de

realizar antes de visitar cada uno de los nodos de la ruta *k*. Los tiempos de desplazamiento de los nodos será la suma de t_{pq} cuando $q>p$

c_k Carga total del vehículo *k.* La carga total del vehículo *k* es la suma de las demandas de los nodos que ha visitado.

CR_k La cabeza de la ruta *k* se corresponde con el nodo en el que está ubicado el vehículo *k* .

D_k Distancia total recorrida por el vehículo *k.* Esta variable es igual a la suma de los desplazamientos entre los nodos que visita este vehículo.

E_k Espera total que realiza el vehículo *k*.

A partir de aquí se establecen el orden de las reglas de decisión aplicadas por el algoritmo y reseñadas previamente en el diagrama de flujo. Si bien el detalle de todos estos procedimientos y funciones aquí sintetizados se presenta en el apartado siguiente.

Generación de lista *L0* de nodos no asignados. Se genera una lista con el número total de nodos distinguiendo entre nodos asignados y no asignados

Inicialización de las *k=1,...,R* rutas a tener en cuenta desde el momento inicial de la planificación, y a todas ellas se les añade el nodo *0* como punto de partida, con los siguientes valores:

$LR_k=[0]$
$CR_k=[0]$
$c_k=0$
$tr_k=0$
$D_k=0$
$E_k=0$

Generación de *L1* de *R* nodos semilla con menores holguras y ordenados de forma ascendente, definida ésta de la siguiente manera: $h_i=tc_i-t_{0i}$ y se selecciona el nodo *i* con $min\ h_i$ Además se computa el valor del menor de los tiempos de servicio.

Si $h_i < min\ ts_i$ se procede a la asignación y adición del nodo *i* a una ruta *k*, y se actualizan sus valores

$LR_k=[0, i]$
$CR_k=[i]$
$c_k=d_i$
$tr_k=t_{oi} + ts_i + te_{ki}$
$D_k=t_{0i}$
$E_k=te_{ki}$

En este caso la espera sería $te_{ki}=[ta_i-(t_{0i})]^+$
Se elimina el nodo *i* de la lista *L1*
Caso contrario, se genera una nueva ruta

Si $h_i \geq min\ ts_i$ Generación de la lista *L2* de nodos *m* de *L0* que se encuentren en camino para ser insertados, tales que:

$$ta_m \le t_{om}$$
$$ta_m + ts_m + t_{mi} \le tc_i$$
$$t_{0m}+ts_m+t_{mi} \le tc_i$$
$$t_{0m}+ts_m+t_{mi}+nh_i = tc_i$$
$$nh_i \ge 0$$
$$t_{0m}+t_{mi} \le \beta \cdot t_{0i}$$
$$t_{0m} + t_{mi} + t_{i0} + te_{ki} \le tc_0$$
$$te_{ki}=[ta_i-[t_{0m}+ts_m+t_{mi})]^+$$
$$d_m+d_i \le q$$

Si la lista *L2* está vacía, entonces la opción es esperar y visitar después de la espera el nodo *i* en consideración

$$LR_k=[0, i]$$
$$CR_k=[i]$$
$$c_k=d_i$$
$$tr_k=t_{0i}+ ts_i + te_{ki}$$
$$D_k=t_{0i}$$
$$E_k=te_{ki}$$

En este caso la espera sería $te_{ki}=[ta_i-(t_{0i})]^+$
Se elimina el nodo *i* de la lista *L1*

Si la lista *L2* no está vacía entonces se genera la lista *L3* de nodos *m* y *l* de *L2* para realizar inserciones dobles. Estos nodos han de ser tales que:

$$ta_m \le t_{0m}$$
$$ta_l \le t_{om}+ts_m+t_{ml}$$
$$t_{0m}+ts_m+ t_{ml}+ts_l+t_{li} \le tc_i$$
$$t_{0m}+ts_m+ t_{ml}+ts_l+t_{li}+ nh'_i = tc_i$$
$$nh'_i \ge 0$$
$$t_{0m}+ t_{ml}+t_{li} \le \gamma \cdot \beta \cdot t_{0i}$$
$$t_{0m} +t_{ml}+ t_{li} + t_{i0} +te_{ki} \le tc_0$$
$$te_{ki}=[ta_i-(t_{0m}+ts_m+t_{ml}+ts_l+t_{li})]^+$$
$$d_m+d_l+d_i \le q$$

Si *L3* está vacía, entonces escogemos el nodo *m* de *L2* que maximiza nh_i, y se añaden a una de las rutas creadas de la siguiente manera:

$$LR_k=[0, m, i]$$
$$CR_k=[i]$$
$$c_k= d_m + d_i$$
$$tr_k=t_{0m} + ts_m+t_{mi}+ts_i+te_{ki}$$
$$D_k= t_{0m} + t_{mi}$$
$$E_k=te_{ki}$$

En este caso $te_{ki}=[ta_i-[t_{0m}+ts_m+t_{mi})]^+$
Se elimina el nodo *i* de la lista *L1*

Se elimina el nodo *m* de la lista *L0*

Si *L3* no está vacía entonces se seleccionan los nodos *m* y *l* que maximizan el valor de nh'_i de manera que se procede a añadir estos dos nodos a una de las rutas originalmente creadas al principio de la planificación de la siguiente manera:

$LR_k=[0, m, l, i]$
$CR_k=[i]$
$c_k= d_m + d_l + d_i$
$tr_k= t_{0m} + ts_m + t_{ml} + ts_l + t_{li}+ts_i+te_{ki}$
$D_k= t_{0m} + t_{ml}+ t_{li}$
$E_k=te_{ki}$

En este caso el $te_{ki}=[ta_i-(t_{0m}+ts_m+t_{ml}+ts_l+t_{li})]^+$
Se elimina el nodo *i* de la lista *L1*
Se eliminan los nodos *m* y *l* de la lista *L0*

Mientras queden nodos en *L0*: Regeneración de *L1* con *R* nuevos nodos. Se seleccionan los *R* nodos con vencimientos de cierre más tempranos ($min\ tc_i$) y se calcula el momento estimado de llegada para cada una de las rutas existentes, así como la holgura resultante para cada ruta *k* y nodo *i* de *L1*.

$$h_i=tc_i-[tr_k+t_{CRk\,i}] \quad \text{para cada ruta } k=1,...,R$$

Se selecciona el nodo *i* con menor tc_i. Para cada nodo *i*, se escoge aquella ruta *k* para la que $min\ t_{CRk\,i}$ y además $h_i \geq 0$

Si $h_i < 0$ Se procede a generar una nueva ruta para el nodo *i*

Si $h_i < min\ ts_i$ Se procede a la adición del nodo *i* a la ruta *k* si se cumplen las siguientes restricciones:

$tr_k+t_{CRk\,i} \leq tc_i$
$tr_k+t_{CRk\,i}+ts_i+ t_{CRk\,0}+te_{ki} \leq tc_0$
$te_{ki}=[ta_i-(tr_k+t_{CR\,i})]^+$
$c_k+d_i \leq q$

Si es así se actualizan los valores de la ruta:

$LR'_k=[LR_k, i]$
$CR'_k=[i]$
$c'_k=c_k + d_i$
$tr'_k=tr_k + t_{CRk\,i}+ ts_i+ te_{ki}$
$D'_k=D_k + t_{CRk\,i}$
$E'_k=E_k +te_{ki}$

En este caso la espera es $te_{ki}=[ta_i - (tr_k + t_{CRk\,i})]^+$

Si estas restricciones no se cumplen, entonces se genera una nueva ruta, y se procede a calcular $R'=R+l$

Si $h_i \geq \min ts_i$ Generación de la lista *L2* de nodos *m* de *L0* que se encuentren en camino para ser insertados, tales que:

$$ta_m \leq tr_k + t_{CRk\,m}$$
$$ta_m + ts_m + t_{mi} \leq tc_i$$
$$tr_k + t_{CRk\,m} + ts_m + t_{mi} \leq tc_i$$
$$tr_k + t_{CRk\,m} + ts_m + t_{mi} + nh_i = tc_i$$
$$nh_i \geq 0$$
$$t_{CRk\,m} + t_{mi} \leq \beta \cdot t_{CRk\,i}$$
$$tr_k + t_{CRk\,m} + ts_m + t_{mi} + ts_i + t_{i0} + te_{ki} \leq tc_0$$
$$te_{ki} = [ta_i - (tr_k + t_{CRk\,m} + ts_m + t_{mi})]^+$$
$$c_k + d_m + d_i \leq q$$

Si la lista *L2* está vacía, entonces la opción es esperar y visitar después de la espera el nodo *i* en consideración. Para ello se actualizan los valores de la ruta *k* en cuestión de la siguiente manera:

$$LR'_k = [LR_k, i]$$
$$CR'_k = [i]$$
$$c'_k = c_k + d_i$$
$$tr'_k = tr_k + t_{CRk\,i} + ts_i + te_{ki}$$
$$D'_k = D_k + t_{CRk\,i}$$
$$E'_k = E_k + te_{ki}$$

En este caso la espera sería $te_{ki} = [ta_i - (tr_k + t_{CRk\,i})]^+$
Se elimina el nodo *i* de la lista *L1*

Si la lista *L2* no está vacía entonces el algoritmo procede a la generación de la lista de posibles inserciones dobles de nodos intermedios, *L3* entre la cabeza de la ruta *k* y el nodo *i* en consideración. Para ello selecciona pares de nodos de *L2* para generar la lista de los nodos a ser insertados simultáneamente. Estos nodos *m* y *l* son tales que cumplen una serie de restricciones, al igual que en la primera etapa. Estas restricciones son las siguientes:

$$ta_m \leq tr_k + t_{CRk\,m}$$
$$ta_l \leq tr_k + t_{CRk\,m} + ts_m + t_{ml}$$
$$tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li} \leq tc_i$$
$$tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li} + nh'_i = tc_i$$
$$nh'_i \geq 0.$$
$$t_{CRk\,m} + t_{ml} + t_{li} \leq \gamma\ \beta \cdot t_{CRk\,i}$$
$$tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li} + ts_i + t_{i0} + te_{ki} \leq tc_0$$
$$te_{ki} = [ta_i - (tr_k + t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li})]^+$$

$$c_k+d_m+d_l+d_i \leq q$$

Si *L3* está vacía, entonces escogemos el nodo *m* de *L2* que maximiza nh_i, y se añaden a una de las rutas inicializadas al principio de la siguiente manera:

$$LR'_k=[LR_k, m, i]$$
$$CR'_k=[i]$$
$$c'_k= c_k + d_m + d_i$$
$$tr'_k=tr_k +t_{CRk\,m}+ ts_m+t_{mi}+ts_i+ te_{ki}$$
$$D'_k= D_k + t_{CRk\,m} + t_{mi}$$
$$E'_k=E_k + te_{ki}$$

En este caso $te_{ki}=[ta_i-(tr_k +t_{CRk\,m} + ts_m + t_{ki})]^+$
Se elimina el nodo *i* de la lista *L1*
Se elimina el nodo *m* de la lista *L0*

Si *L3* no está vacía entonces se seleccionan los nodos *m* y *l* que maximizan el valor de nh'_i de manera que se procede a añadir estos dos nodos a la ruta *k* de la siguiente manera:

$$LR'_k=[LR_k, m, l, i]$$
$$CR'_k=[i]$$
$$c'_k= c_k + d_m + d_l + d_i$$
$$tr'_k=tr_k+ t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li}+ts_i+te_{ki}$$
$$D'_k=D_k+ t_{CRk\,m} + t_{ml}+ t_{li}$$
$$E'_k=E_k + te_{ki}$$

En este caso el $te_{ki}=[ta_i-(tr_k +t_{CRk\,m} + ts_m + t_{ml} + ts_l + t_{li})]^+$
Se elimina el nodo *i* de la lista *L1*
Se eliminan los nodos *m* y *l* de la lista *L0*

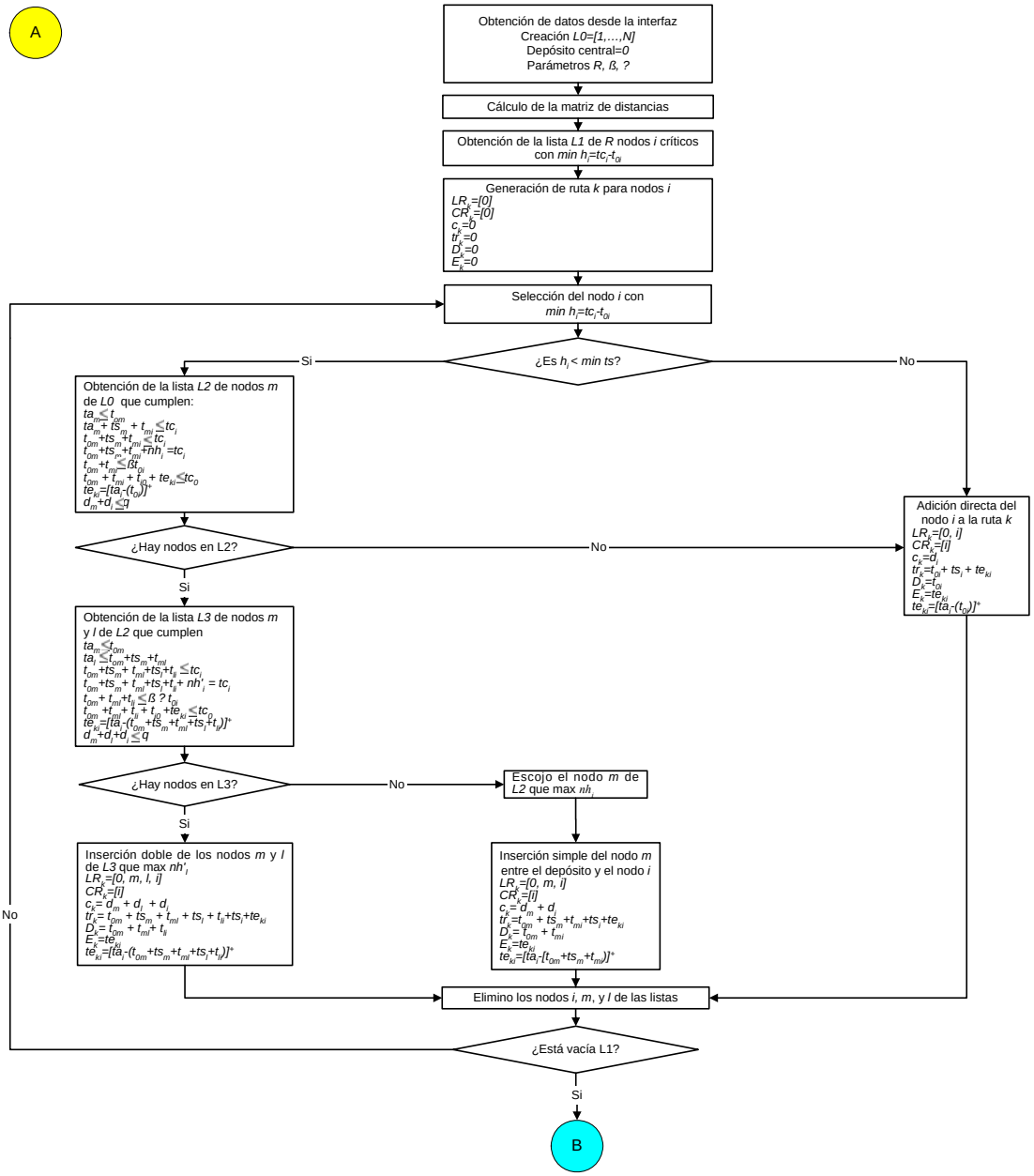
Retorno de vehículos al nodo *0* Cuando no queden nodos sin asignar en *L0*, se procede al retorno de los vehículos y se actualizan todos los valores de las rutas procediendo a los siguientes cálculos para la cualquier ruta *z*:

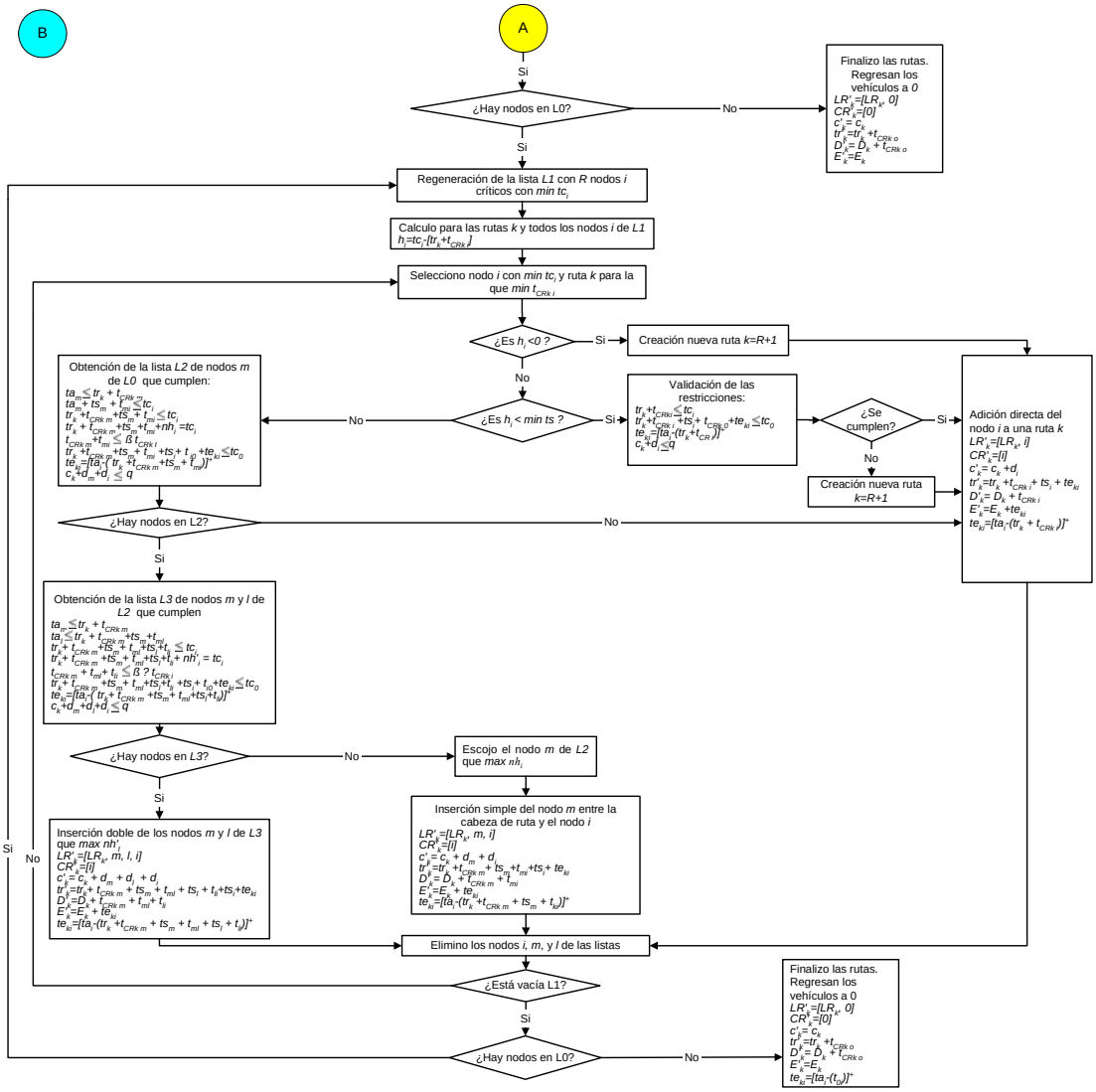
$$LR'_k=[LR_k, 0]$$
$$CR'_k=[0]$$
$$c'_k= c_k$$
$$tr'_k=tr_k +t_{CRk\,o}$$
$$D'_k= D_k + t_{CRk\,o}$$
$$E'_k=E_k$$

Finalmente el algoritmo genera una tabla con los datos resumen de cada una de las rutas, y los datos globales de la solución.

1.9 Diagrama de flujo del algoritmo

A continuación se recoge un diagrama de flujo del algoritmo desarrollado, y que sirve de guía para la programación efectuada, de la que se han incluido algunos fragmentos en el anexo II para facilitar posteriores desarrollos.


A
Obtención de datos desde la interfaz
Creación L0=[1,…,N]
Depósito central=0
Parámetros R, ß, ?
Cálculo de la matriz de distancias
Obtención de la lista L1 de R nodos i críticos con min h_i=tc_i-t_0i
Generación de ruta k para nodos i
LR_k=[0]
CR_k=[0]
c_k=0
tr_k=0
D_k=0
E_k=0
Selección del nodo i con min h_i=tc_i-t_0i
¿Es h_i < min ts?
Si
No
Obtención de la lista L2 de nodos m de L0 que cumplen:
ta_m ≤ t_0m
ta_m + ts_m + t_mi ≤ tc_i
t_0m +ts_m +t_mi ≤ tc_i
t_0m +ts_m +t_mi + Fnh'_i =tc_i
t_0m +t_mi ≤ ßt_0i
t_0m + t_mi + t_i0 + te_ki ≤ tc_0
te_ki=[ta_i-(t_0i)]+
d_m+d_i ≤ q
¿Hay nodos en L2?
No
Si
Adición directa del nodo i a la ruta k
LR_k=[0, i]
CR_k=[i]
c_k=d_i
tr_k=t_0i + ts_i + te_ki
D_k=t_0i
E_k=te_ki
te_ki=[ta_i-(t_0i)]+
Obtención de la lista L3 de nodos m y l de L2 que cumplen
ta_m ≤ t_0m
ta_l ≤ t_0m +ts_m +t_ml
t_0m +ts_m + t_ml +ts_l +t_li ≤ tc_l
t_0m +ts_m + t_ml +ts_l +t_li + nh'_l = tc_l
t_0m + t_ml +t_li ≤ ß ? t_0l
t_0m +t_ml + t_li + t_l0 +te_ki ≤ tc_0
te_ki=[ta_i-(t_0m +ts_m +t_ml +ts_l +t_li)]+
d_m+d_l+d_i ≤ q
¿Hay nodos en L3?
No
Si
Escojo el nodo m de L2 que max nh_i
Inserción doble de los nodos m y l de L3 que max nh'_i
LR_k=[0, m, l, i]
CR_k=[i]
c_k= d_m + d_l + d_i
tr_k= t_0m + ts_m + t_ml + ts_l + t_li +ts_i+te_ki
D_k= t_0m + t_ml+ t_li
E_k=te_ki
te_ki=[ta_i-(t_0m+ts_m+t_ml+ts_l+t_li)]+
Inserción simple del nodo m entre el depósito y el nodo i
LR_k=[0, m, i]
CR_k=[i]
c_k= d_m + d_i
tr_k=t_0m + ts_m +t_mi +ts_i +te_ki
D_k= t_0m + t_mi
E_k=te_ki
te_ki=[ta_i-(t_0m+ts_m+t_mi)]+
Elimino los nodos i, m, y l de las listas
¿Está vacía L1?
No
Si
B


B
A
Si
¿Hay nodos en L0?
No
Finalizo las rutas. Regresan los vehículos a 0
Regeneración de la lista L1 con R nodos i críticos con min tc_i
Calculo para las rutas k y todos los nodos i de L1
Selecciono nodo i con min tc_i y ruta k para la que min t_CRk i
¿Es h_i <0 ?
Creación nueva ruta k=R+1
Obtención de la lista L2 de nodos m de L0 que cumplen:
¿Es h_i < min ts ?
Validación de las restricciones:
¿Se cumplen?
Creación nueva ruta k=R+1
Adición directa del nodo i a una ruta k
¿Hay nodos en L2?
Obtención de la lista L3 de nodos m y l de L2 que cumplen
¿Hay nodos en L3?
Escojo el nodo m de L2 que max nh_i
Inserción simple del nodo m entre la cabeza de ruta y el nodo i
Inserción doble de los nodos m y l de L3 que max nh'_i
Elimino los nodos i, m, y l de las listas
¿Está vacía L1?
¿Hay nodos en L0?
Finalizo las rutas. Regresan los vehículos a 0

Capítulo 4

Experimentación y Resultados

En el presente capítulo se muestran los resultados obtenidos en la fase de experimentación a través de la implementación del algoritmo desarrollado en el capítulo anterior. Partiendo de la revisión de los métodos utilizados en la literatura para la resolución del VRPTW y el propio método desarrollado en este trabajo, se presentan diferentes comparativas de la bondad de unos y otro. Para ello se ha dividido el capítulo en tres grandes apartados. En primer lugar se presenta la aplicación del método desarrollado en este trabajo a los problemas tipo de Solomon (1987) para contrastar su bondad de ajuste, así como su robustez. Es necesario tener en cuenta que el método propuesto no persigue superar los récords de los algoritmos ya desarrollados en la literatura, puesto que en ese sentido se hubiera optado por líneas de computación exhaustiva, marcadas por el desarrollo de métodos metaheurísticos, o incluso por la aplicación de técnicas de resolución exacta. Recordamos que la finalidad del algoritmo propuesto es arrojar más luz sobre el comportamiento de las reglas de decisión en la construcción de rutas. Igualmente, se realiza un análisis de sensibilidad de los parámetros del problema para obtener mejores ajustes y tiempos de computación, estableciendo las particularidades de cada uno de los tipos de problema propuestos por Solomon. En segundo lugar, se presentan los resultados de la aplicación de los métodos más conocidos presentados por los diversos autores a los problemas tipo, tal y como marcan las pautas en esta línea de investigación. Para ello se presenta una tabla comparativa de los diferentes métodos, divididos en los diferentes subgrupos contemplados en la revisión literaria realizada (i.e. métodos de construcción, métodos de mejora, y métodos metaheurísticos). En tercer lugar se recoge una sección con las conclusiones obtenidas en esta fase de investigación, destacando las ventajas e inconvenientes del método desarrollado.

1 Introducción

La experimentación desarrollada durante el curso del presente trabajo tiene como finalidad el análisis de las reglas de decisión de los algoritmos presentados en la literatura con el objetivo de comprender y aprender, tanto de estas reglas de decisión utilizadas, como de la propia complejidad de los problemas tipo para de esta manera sentar las bases y fundamentos para el desarrollo de un método de construcción de rutas que sea eficiente en cuanto a los tiempos de computación y eficaz, en cuanto a los resultados obtenidos. Es por esto que se persigue aprender de todos y cada uno de los métodos buscando en todo momento la crítica de las reglas aplicadas, y procurando en la medida de lo posible de buscar una lógica a estas reglas que sean comparables con las decisiones que aplican los planificadores empresariales. Generalmente en la literatura nos encontramos con algoritmos que se fundamentan únicamente en métodos matemáticos de aproximación, donde se aplican todo tipo de procedimientos y mecanismos para mejorar soluciones, basados en búsquedas intensivas, en muchos casos aleatorias. Este tipo de algoritmos aportan poca información sobre cómo se han de estructurar las decisiones para aplicar en los procesos de planificación de rutas. Por otro lado, nos encontramos con algunos algoritmos donde sí existen reglas lógicas que se asemejan a aquellas que aplicaría un decisor en una empresa y aunque son peores que los métodos de tipo computacional, son realmente los que nos interesan en este trabajo.

La experimentación se llevó a cabo a través de la implementación del algoritmo desarrollado en este trabajo sobre los problemas tipo de la literatura. En este sentido, dos versiones del mismo fueron implementadas sobre toda la batería de problemas de Solomon (1987), con diferentes resultados. La primera versión considera como nodos a ser insertados aquellos que estaban más cerca del nodo destino (versión *A*), mientras que la segunda versión considera como nodos a ser insertados aquellos que minimizan el desplazamiento adicional provocado (versión *B*). A pesar de que la versión *B* arroja normalmente mejores resultados, existen casos en los que es preferible la aplicación del criterio de inserción contemplado en la versión *A.* Por ello se recogen en la siguiente tabla éstas diferencias para ambas versiones. A pesar de estas diferencias, en el presente trabajo solamente se desarrolla la versión *B,* por ser la que normalmente se comporta mejor.

	Versión A		Versión B	
Problema	**Distancia**	**Nº vehículos**	**Distancia**	**Nº vehículos**
R103	1729	20	1742	23
R110	1688	22	1706	23
C103	1362	11	1381	11
C108	1801	24	1819	25
RC102	2192	24	2217	25
RC104	1678	14	1696	14
R205	1435	12	1471	11

Tabla 4.1 Comparación de resultados de las dos versiones del algoritmo

1.1 Recursos utilizados

Para la implementación del algoritmo se utilizó inicialmente un ordenador portátil *ACER Extensa 367D*, con procesador *Pentium II* a 200 MHz. En este ordenador el tiempo de computación ascendía aproximadamente a 55 soluciones individuales por segundo (aproximadamente 0,02 segundos cada solución individual), resultando por ello enormemente rápido en el cálculo individual de resultados. Sin embargo en la aplicación del barrido de posibles combinaciones que también recoge el algoritmo (recordamos que ascendía a 4.000.000 búsquedas individuales) este ordenador se presentaba excesivamente lento, dado que por término medio tardaba un total de 72.000 segundos por cada uno de los problemas tipo de Solomon, lo que equivalía a tiempos de computación de unas 20 horas por cada problema. Por ello se optó por recurrir a un ordenador con un procesador mucho más rápido. En este sentido se utilizó en segundo lugar un ordenador *Toshiba* con procesador *Pentium IV* a 1,7 GHz. En este ordenador los tiempos de computación se reducían drásticamente, hasta el punto de realizar esos 4.000.000 iteraciones en un tiempo estimado medio de unos 9.000 segundos. Sin embargo la implementación a los 56 problemas seguía suponiendo elevados tiempos de computación para un único ordenador, por lo que finalmente se optó por buscar recursos adicionales, por lo que se recurrió a la utilización de dos clusters de ordenadores donde se implementó de forma individual en cada ordenador el algoritmo en cuestión para establecer los cálculos relativos a los 56 problemas tipo. Es necesario destacar que no llevó a cabo ningún tipo de computación paralela, ya que todos los cálculos se realizaron de forma independiente en cada ordenador. Estos ordenadores constaban de procesadores *AMD Duron* a 800 MHz, más rápidos que el ordenador original. Los clusters contenían 10 ordenadores uno, y 20 ordenadores el segundo, ascendiendo a un total de 30 ordenadores. En este procesador los tiempos medios para el barrido, ascendían a unas 12 horas aproximadamente de manera que los tiempos totales de computación invertidos en el cálculo de los resultados presentados en este trabajo ascendieron aproximadamente a 800 horas de computación. A continuación se muestran las velocidades de cálculo de los diferentes procesadores utilizados, expresadas en Mflops/seg.

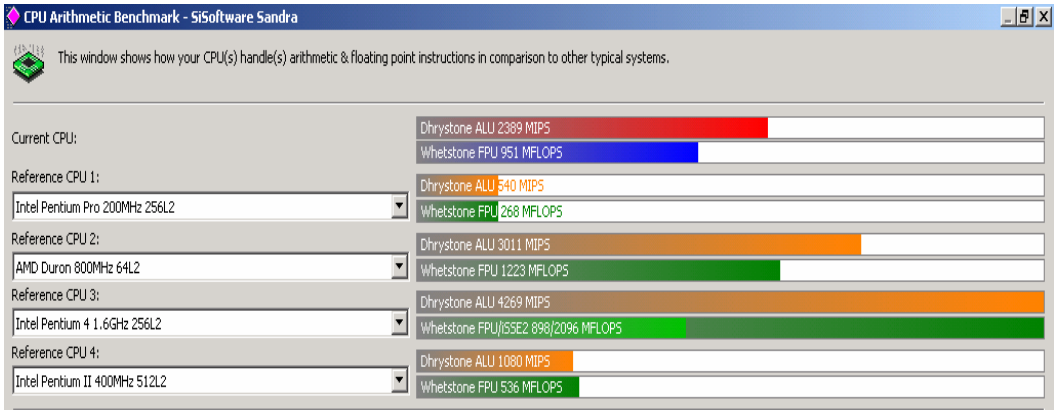


Tabla 4.2 Comparación de las velocidades de computación medidas en Mflops,/s de los procesadores utilizados durante la investigación, según el programa SANDRA

1.2 Características de los problemas tipo

Recordamos también en este punto las principales características de los 56 problemas presentados por Solomon (1987) como punto de partida para el desarrollo de esta línea de investigación[20].

Cada uno de los 56 problemas tipo cuenta con un total de 100 clientes, de manera que el tiempo de desplazamiento entre cada uno de estos clientes es coincidente con la distancia euclídea entre los mismos. Además todos los problemas cuentan con un depósito central del que han de partir los vehículos en consideración. Los datos se corresponden con diferentes situaciones de los clientes, así como diferencias temporales entre las ventanas de tiempo consideradas, y las propias demandas de los clientes.

Cada uno de los problemas contempla siempre una flota de 25 vehículos con capacidades limitadas y homogéneas.

Los problemas se han dividido en 2 categorías según la dispersión geográfica de los clientes a ser servidos. En primer lugar nos encontramos con los problemas de tipo *R*, que consisten en ejemplos donde los clientes están dispersos aleatoriamente en el plano a través de una distribución uniforme. Los problemas de tipo *C* contemplan una disposición de los clientes en grupos geográficamente dispersos en el plano. En este caso los clientes están clusterizados. Por último nos encontramos con los problemas de tipo *RC*, donde se presenta una mezcla de los dos casos anteriores.

Para cada una de las categorías anteriores se presentan dos conjuntos de problemas que se denotan por los subíndices *1* y *2*. de manera que en cada uno de los dos subgrupos se presentan entre 8 y 12 problemas individuales.

Los problemas relativos al conjunto *1*, tienen un horizonte de planificación muy breve, de manera que son necesarios múltiples vehículos para realizar los servicios correspondientes a todos los clientes, de forma que cada ruta considerada debe realizar pocas visitas, requiriendo por lo tanto un elevado número de rutas.

Los problemas relativos al conjunto *2* son problemas con horizontes de planificación más amplios, de manera que no son necesarios tantos vehículos como en el caso anterior. Generalmente se trata de problemas cuya solución integra pocas rutas, y éstas son muy largas, incluyendo muchas vistas.

Las diferencias entre estos problemas radican en que las demandas de cada uno de los clientes, así como las ventanas de tiempo para los mismos, son distintas de manera que en cada uno de los subgrupos existen entre 8 y 12 problemas diferentes. Otra consideración importante es que dentro de todos los subconjuntos la duración de los tiempos de servicio siempre es igual.

[20] Disponibles en http://www.idsia.ch/~luca/macs-vrptw/problems/

A partir de la propuesta de estos problemas tipo, la mayor parte de los autores contrastan la eficacia de sus métodos contra las mejores soluciones halladas hasta el momento, de manera que cada año se mejoran las soluciones de partida. Estos resultados se mantienen actualizados en multitud de sitios Web, para consulta de los investigadores. Son estos mismos resultados los que se han utilizado como referencia para el presente trabajo, tal y como se recogen en el anexo V.

2 Análisis de resultados proporcionados por el algoritmo

En este apartado se comentan los resultados obtenidos en la aplicación del algoritmo desarrollado en el marco del presente trabajo a los problemas tipo presentados por Solomon (1987). Para ello se ha optado por aportar los resultados de cada uno de los problemas en cuestión, tal y como los ha obtenido el algoritmo, y agrupados por categorías. La finalidad de este estudio particularizado del comportamiento de este método en la resolución de cada uno de los problemas obedece a la necesidad de arrojar más luz sobre las características intrínsecas del algoritmo desarrollado, el comportamiento de los parámetros seleccionados en cada caso, así como el correspondiente análisis de sensibilidad de los mismos, y el mayor conocimiento de los problemas propuestos por Solomon (1987).

Merece la pena recordar en este punto que para cada uno de los problemas se han probado un total de 4.000.000 de combinaciones diferentes de parámetros según los siguientes intervalos de variación:

R	Entre 1 y 25, con incrementos unitarios, (ambos incluidos)
β	Entre 1 y 5, con incrementos centesimales.
γ	Entre 1 y 5, con incrementos centesimales.

Por esto, el total de combinaciones posibles asciende a 4.000.000 de combinaciones posibles, de manera que los resultados aportados en este capítulo corresponden a las mejores combinaciones obtenidas para estos parámetros, y para cada uno de los problemas en cuestión.

2.1 Grupo de problemas R1

En el caso de los problemas *R1* recordamos que constituyen un conjunto de ejemplos donde las rutas son relativamente cortas, por lo que es necesario utilizar un elevado número de vehículos para realizar los servicios. Igualmente, los clientes están dispersos aleatoriamente en el espacio, y tienen ventanas de tiempo muy restrictivas. La topología de estos problemas se corresponde con la figura 4.1:

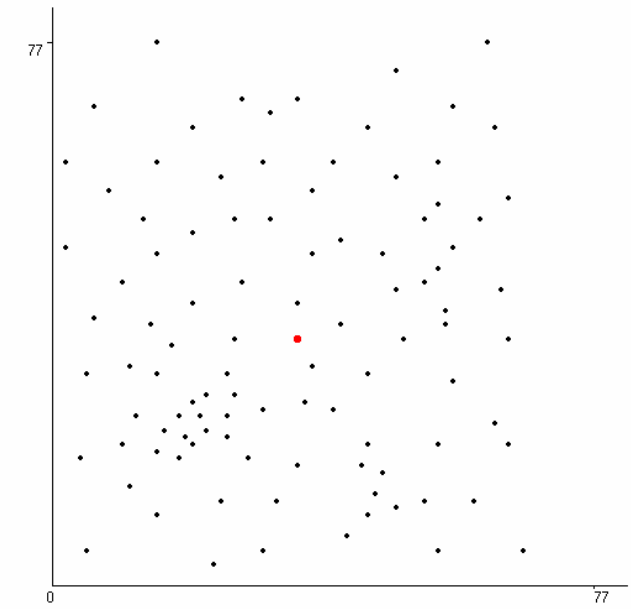


Figura 4.1 Topología clásica de los problemas tipo *R*

Para este conjunto de problemas las soluciones obtenidas por el algoritmo son las siguientes:

	R	***β***	***γ***	**Distancia**	**NV**	**Mejor Resultado**	**Exceso en %**
R101	13	1,00	1,00	2805	46	1650,00	70%
R102	8	1,00	1,00	2549	37	1486,12	72%
R103	20	1,74	1,06	1742	23	1292,85	35%
R104	5	1,16	1,00	1326	14	982,01	35%
R105	8	1,00	1,00	2682	40	1377,11	95%
R106	4	1,10	1,00	1842	23	1252,03	47%
R107	1	1,10	1,12	1477	17	1113,69	33%
R108	11	1,02	1,01	1324	15	964,38	37%
R109	8	1,00	1,00	2892	38	1194,73	142%
R110	16	1,49	1,00	1706	23	1124,40	52%
R111	3	1,34	1,16	1602	19	1096,72	46%
R112	18	1,40	1,00	1517	18	1003,73	51%
Acumulado	115	14,35	12,35	23464	313	14537,77	61%
Media	9,58	1,20	1,03	1955,33	26,08	1211,48	61%

Tabla 4.3 Resultados para los problemas tipo *R1*

En la tabla 4.3 se muestran los resultados para el conjunto de problemas *R1*. En la cabecera se muestran los datos del número inicial de rutas seleccionado, así como los valores de los parámetros β y γ mejores para cada uno de los casos. La solución se compone del valor relativo a la distancia total recorrida por los vehículos, así como el

número de rutas integrantes de la solución. Las condiciones de este tipo de problemas son tales que bien por las ventanas de tiempo, o bien por las limitaciones de carga de los vehículos, existe un elevado número de rutas. No se ha incluido el dato relativo al número de vehículos de las mejores soluciones, dado que el objetivo último consiste en la minimización de la distancia total recorrida. Sin embargo estos datos se recogen en la tabla presentada en el anexo V, al igual que se precisa la autoría de estos mejores resultados. Igualmente, en las tablas comparativas con otros métodos, sí se recogen estos datos para que el lector pueda forjarse una idea de las diferencias entre unos y otros.

Número inicial de rutas, o nodos semilla.

En este tipo de problemas cabe destacar la conveniencia de comenzar por un número reducido de rutas, incluso menor que el número mínimo marcado por las restricciones de carga, y que resulta un indicador de un límite inferior para este tipo de problemas. En este sentido es el propio algoritmo quien va generando rutas a posteriori según las necesidades del problema, de manera que el número de rutas finales supone por término medio un 170% de las rutas iniciales consideradas por el usuario. Es necesario tener en cuenta que en el algoritmo desarrollado, y a diferencia de otros métodos de resolución propuestos en la literatura, el objetivo principal es la minimización de la distancia total recorrida, y no del número de vehículos utilizados, por lo que el valor de esta variable en la solución solamente ha de ser tenido en cuenta a efectos ilustrativos de la operativa del algoritmo.

Parámetro β

En el caso del parámetro β nos encontramos con que su valor resulta trascendente para este tipo de problemas. Recordamos que el parámetro β constituía un limitador de las distancias adicionales resultantes de posibles inserciones de nodos entre la cabeza de cada ruta, y el nodo crítico a ser añadido a la misma. En este sentido, existe una limitación en toda inserción que viene marcada por las ventanas temporales, y las holguras resultantes, tal y como se recoge en el capítulo anterior. Sin embargo, este limitador ha tomado valores reducidos, de manera que por término medio solamente se admiten incrementos de distancia en las inserciones, de un 20% aproximadamente sobre la distancia original entre nodos, rechazando posibles inserciones que bien podrían ser válidas por cuanto respetarían las restricciones temporales, pero que en general empeorarían la solución final. Es decir, que es preferible discriminar algunas inserciones posibles a consta de posteriores visitas, y de esta manera solamente admitir inserciones que estén próximas a la línea recta.

Parámetro γ

En el caso del parámetro multiplicador de distancias para dobles inserciones entre nodos, cabe destacar la baja importancia del mismo por término medio en este tipo de problemas. Si bien, al estar combinado con el valor del parámetro β, la amplitud del área la determina en mayor medida el parámetro β, aunque en algún caso ha sido necesario ampliar

artificialmente esta área a través del parámetro γ para obtener un mayor número de posibilidades en las inserciones dobles.

	Nº Ventanas Bilaterales	Nº de Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
R101	100	0	10	-	10
R102	75	25	10	Solamente tc	10
R103	50	50	10	Solamente tc	10
R104	25	75	10	Solamente tc	10
R105	100	0	30	-	10
R106	75	25	30	Solamente tc	10
R107	50	50	30	Solamente tc	10
R108	25	75	30	Solamente tc	10
R109	100	0	Variables	Variables	10
R110	75	25	Variables	Variables	10
R111	50	50	Variables	Variables	10
R112	Variables	Variables	Variables	Variables	10

Tabla 4.4 Composición de los problemas *R1*

El comportamiento del algoritmo es mejor en aquellos casos donde existe un equilibrio entre el número de nodos sujeto a ventanas de tiempo bilaterales, y otros nodos sin restricciones temporales tan fuertes. Esta afirmación es un tanto lógica si atendemos al funcionamiento del algoritmo. En primer lugar cabe decir que la existencia de un número de ventanas de tiempo no muy elevado, (inferior al 75% del total) no es excesivamente perjudicial para el algoritmo, dado que si bien, estas ventanas suponen mayores restricciones al problema, también sirven de ayuda al algoritmo, para seleccionar aquellas que sí son factibles. Por el contrario, los algoritmos de búsqueda aleatoria suelen tener un mayor número de problemas, ya que muchas de sus combinaciones de nodos o arcos son rechazas por violar las restricciones del problema. En este caso ocurre precisamente lo contrario, puesto que sirven de guía a las decisiones adoptadas por el algoritmo. Por esto podemos afirmar que el comportamiento en el caso de los problemas *R1* donde el número de ventanas de tiempo estrechas no es excesivo, el incremento medio de la distancia total recorrida, con respecto a los mejores resultados hallados hasta el momento es de un 45% aproximadamente, no suponiendo un inconveniente ni el tamaño de las ventanas de tiempo, ni la variabilidad de éstas para los nodos considerados. De ahí que en los problemas R102, R103, R104, R106, R107, R108 el comportamiento del algoritmo sea mejor que en los problemas R101 y R105, donde el número de nodos con ventanas de tiempo bilaterales es del 100%, no existiendo nodos libres para que funcione el mecanismo de inserción.

Igualmente podemos afirmar que el incremento sobre la distancia total recorrida se reduce sensiblemente a medida que se incrementa la flexibilidad de los nodos posibles para ser insertados. Recordemos que el algoritmo se guía por los momentos de cierre de los nodos del problema, estableciendo posibles inserciones entre los nodos críticos seleccionados. De esta manera cuanto mayor es el número de nodos libres, mejores son las opciones de inserción, y menor es la distancia total recorrida.

El problema surge cuando todos los nodos del problema están sujetos a ventanas de tiempo estrechas, de manera que los resultados empeoran. En este sentido nos encontramos con que generalmente no es posible realizar ningún tipo de inserción, ya que la programación de las rutas obedece estrictamente a cuestiones temporales de los nodos. De ahí la razón de que para los problemas R101, R105 y R109, que son aquellos para los que el 100% de los nodos tienen ventanas de tiempo estrechas, el algoritmo no considera el mecanismo de inserción, por lo que los parámetros β y γ son equivalentes a la unidad, rechazando toda posibilidad de inserción. En este sentido la elección de los nodos semilla es la gran baza para establecer la mejor solución, así como las ulteriores uniones de nodos a las rutas preestablecidas, contemplando o no posibles esperas, ante la posibilidad de recorrer amplias distancias.

2.2 Grupo de problemas R2

En el caso de los problemas del tipo *R2* nos encontramos con la misma topología que en los problemas *R1*, donde los clientes están repartidos de forma aleatoria en el plano. Sin embargo, a diferencia de los problemas *R1*, en este caso las rutas son largas, de manera que la necesidad de vehículos disminuye drásticamente, y las rutas se alargan en el tiempo y en el espacio. De esta manera el horizonte temporal del depósito central también se amplía, permitiendo un mayor número de visitas de los vehículos. Igualmente se incrementa la capacidad de los vehículos permitiendo múltiples cargas en los diferentes clientes visitados.

	R	***β***	***γ***	**Distancia**	**NV**	**Mejor Resultado**	**Exceso en %**
R201	7	1,80	1,00	1882	22	1252,37	50%
R202	6	1,85	1,05	1377	6	1191,70	16%
R203	4	2,03	1,16	1160	4	942,64	23%
R204	4	1,47	1,00	953	4	849,62	12%
R205	10	1,94	1,36	1471	11	994,42	48%
R206	6	1,29	1,14	1163	6	912,97	27%
R207	5	1,25	1,06	1021	5	914,39	12%
R208	4	1,50	1,02	899	5	731,23	23%
R209	10	2,639	1,00	1346	12	909,86	48%
R210	6	1,46	1,00	1178	6	955,39	23%
R211	9	1,49	1,00	1179	9	910,09	30%
Acumulado	71,00	18,72	11,79	13629	90	10565	29%
Media	6,45	1,70	1,07	1239	8	960	29%

Tabla 4.5 Resultados para los problemas tipo *R2*

Número inicial de rutas, o nodos semilla

En este caso nos encontramos con que el número de nodos iniciales o nodos semilla es un valor bastante reducido por término medio, de manera que es posible afirmar que en este caso, y como cabría pensar de antemano es preferible comenzar con un valor reducido de rutas, cercanas generalmente a las rutas de la solución final. Si observamos está comparativa, nos encontramos que por término medio se debe comenzar la planificación con un valor de nodos semilla de entre 6 y 7, y que finalmente nos encontraremos con un número de rutas que por término medio es de 8, por lo que no suelen aparecer nuevas rutas en el curso de la planificación, de forma que la importancia en la selección de este parámetro no es tan elevada como en el caso de los problemas R1.

Parámetro β

Con respecto a los valores del parámetro β, nos encontramos con que por término medio, se suelen considerar amplias zonas de influencia marcadas por β de 1.7, y que en algunos casos sobrepasan al 200% de las distancias originales entre los nodos en consideración. En este sentido cabe destacar la necesidad de ampliar las zonas de inserción. En este sentido, generalmente los valores de los nodos semilla aportan poca información, dado que generalmente estos nodos semilla son los que darán lugar a las rutas de la solución final, salvo en algunas excepciones como R201. Por ello la consideración de que estos parámetros de control de las zonas de influencia van a jugar un papel relevante en las grandes distancias, siendo el resultado final, una mayor permisividad de inserciones en este tipo de problemas que en el caso de los problemas *R1*. Por ello en general una mayor zona de influencia para proceder a inserciones de nodos, suele ir asociada a un menor número de rutas, siempre que las restricciones temporales lo permitan, de manera que las esperas ante los nodos críticos en cuestión son suplidas por visitas intermedias de nodos que están abiertos.

Parámetro γ

En este tipo de problemas, también juega un papel importante los valores del parámetro γ, ya que ocasionalmente incrementa la zona de inserción en el caso de considerar dobles inserciones entre los nodos críticos y las rutas. Generalmente, los valores de γ son superiores que en otros casos, de manera que los vehículos aprovechan los tiempos de espera antes de las aperturas de los nodos más críticos, con visitas a nodos que se encuentren en el camino, para no tener que realizar estas visitas al regreso al depósito.

En el grupo de problemas de tipo *R2* nos encontramos con una gran variedad de ejemplos, que podemos agrupar en tres subclases. Los cuatro primeros se caracterizan por tener tamaños variados de las ventanas de tiempo, y diferentes proporciones del número de ventanas bilaterales, sobre las ventanas unilaterales. En el segundo grupo nos encontramos con tamaños de ventanas de tiempo homogéneos de 240, combinando diferentes proporciones de nodos con ventanas de tiempo bilaterales, y nodos con ventanas de tiempo unilaterales. Por último, en el tercer grupo nos encontramos con una

mezcla en los tamaños de las ventanas de tiempo, y diferentes composiciones de las carteras de clientes, según tengan más o menos ventanas de tiempo bilaterales o unilaterales.

	Nº Ventanas Bilaterales	Nº de Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
R201	100	0	Variado	-	10
R202	75	25	Variado	Solamente tc	10
R203	50	50	Variado	Solamente tc	10
R204	25	75	Variado	Solamente tc	10
R205	100	0	240	-	10
R206	75	25	240	Solamente tc	10
R207	50	50	240	Solamente tc	10
R208	25	75	240	Solamente tc	10
R209	Grande	Reducido	Variables	Variables	10
R210	Intermedio	Intermedio	Variables	Variables	10
R211	Pequeño	Grande	Variables	Variables	10

Tabla 4.6 Composición de los problemas *R2*

Para los problemas del grupo *R2*, y al igual que ocurría en los problemas de tipo R1, podemos afirmar que salvo en los casos donde el 100% de los clientes tienen ventanas bilaterales, y que se corresponden con los problemas R201, R205 y R209, nos encontramos con que en general el algoritmo aporta buenos resultados de manera que el incremento que por término medio se obtiene con respecto a los mejores resultados es de aproximadamente un 20%, siendo globalmente por término medio un 30% peor que los mejores resultados obtenidos en este tipo de problemas. Esta conclusión también es bastante lógica por cuanto es necesario tener nodos libres para insertar, de manera que si todos los nodos están sujetos a ventanas de tiempo bilaterales y estrechas, entonces no existirán nodos libres para poder ser insertados entre los nodos seleccionados como críticos por el algoritmo, y las rutas a las que han sido asignados.

También se puede observar este fenómeno en el número de vehículos necesarios para los problemas donde existen mayores restricciones temporales, de manera que tanto el número de clientes semillas, como el número final de las rutas es mayor para estos casos (R201, R205, y R209)

En este tipo de problemas, la no coincidencia del tamaño de las ventanas de tiempo con los tiempos de servicio, permite las inserciones de nodos, aún estando todos ellos restringidos por las mismas, algo que no ocurría en el caso de los problemas de tipo *R1*. En este sentido, y tal y como se apuntaba anteriormente, la importancia de los parámetros de control de la zona de influencia adquiere un grado de implicación más elevado que en caso anterior.

2.3 Grupo de problemas RC1

En el caso de los problemas de Solomon tipo *RC1* nos encontramos con una combinación entre los tipos *R1* y *C1*, consistente en que los clientes del mismo están parcialmente

clusterizados, pero al mismo tiempo existe otro conjunto de nodos que se han dispuesto de forma aleatoria. En el caso de los problemas tipo *1*, la duración del horizonte de planificación es breve, e igualmente las restricciones temporales impuestas por los clientes obligan a realizar múltiples rutas.

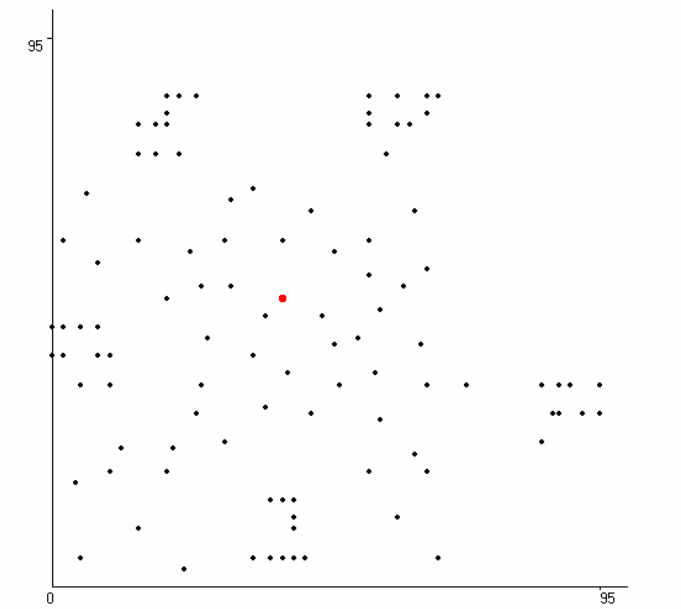


Figura 4.2 Topología clásica de los problemas tipo *RC*

Las soluciones para este tipo de problemas se presentan en el siguiente cuadro.

	R	***β***	***γ***	**Distancia**	**NV**	**Mejor Resultado**	**Exceso en %**
RC101	9	1,00	1,00	3163	38	1696,94	86%
RC102	8	1,24	1,00	2217	23	1554,75	43%
RC103	9	1,23	1,01	1892	18	1262,02	50%
RC104	11	2,71	1,74	1696	14	1135,48	49%
RC105	6	1,26	1,01	2259	25	1633,72	38%
RC106	9	1,00	1,00	2787	32	1427,13	95%
RC107	4	2,00	1,00	2020	22	1230,54	64%
RC108	19	1,50	1,00	1940	20	1139,82	70%
Acumulado	75,00	11,94	8,76	17974	169	11080	62%
Media	9,38	1,49	1,10	2247	24	1385	62%

Tabla 4.7 Resultados de los problemas tipo *RC1*

Número de rutas iniciales, o nodos semilla

Generalmente el número de nodos semilla es un factor decisivo a la hora de establecer su valor. En este caso nos encontramos con que la media de nodos con los que se ha de comenzar a planificar está entre 9 y 10, pero a medida que avanza la planificación, este

número se incrementa considerablemente hasta alcanzar una medida de 21 nodos en las soluciones finales. En este sentido la complejidad de este tipo de problemas es semejante a lo que se exponía para el caso de los problemas *R1*, ya que tanto la topología de los nodos, como las restricciones temporales son muy similares.

Parámetro β

En este caso los valores del parámetro β son cercanos a 1.5, de manera que podemos afirmar que la importancia del mecanismo de inserción para estos ejemplos es elevada, algo mayor que en el caso de los problemas *R1*. Esta afirmación también tiene una cierta lógica evidente si pensamos en que en el caso de los problemas *R1*, los nodos se distribuían de forma aleatoria en el plano, e uniforme, de manera que las distancias entre cada par de nodos eran más o menos homogéneas. Por ello las zonas de inserción tampoco necesitaban ser excesivamente amplias. Sin embargo en el caso de los problemas donde los nodos están más agrupados o clusterizados, es necesario ampliar más la zona de inserción para encontrar candidatos a ser insertados entre la cabeza de una ruta, y el nodo crítico en cuestión. Por esto en este caso el valor de β es más amplio que en el caso de los problemas *R1*.

Parámetro γ

Lo mismo ocurre con el valor de γ, como multiplicador de esta zona de inserción, que provoca que se amplíe para contemplar las distancias resultantes en inserciones dobles de nodos. También en este caso se puede hacer la misma afirmación que para el caso del parámetro β, ya que una vez más el valor de γ para los problemas *RC1* es superior al valor en el caso de los problemas *R1*.

	Nº Ventanas Bilaterales	Nº Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
RC101	100	0	30	-	10
RC102	75	25	30	Solamente tc	10
RC103	50	50	30	Solamente tc	10
RC104	25	75	30	Solamente tc	10
RC105	100	0	Entre 10 y 140	-	10
RC106	100	0	60	-	10
RC107	100	0	Entre 50 y 140	-	10
RC108	100	0	Entre 30 y 140	-	10

Tabla 4.8 Composición de los problemas *RC1*

Tras el análisis de las características individuales de cada problema, podemos afirmar que al igual que ocurría con los problemas de tipo *R1*, el algoritmo se comporta mejor en aquellos casos donde no todos los nodos tienen ventanas rígidas de tiempo, como es el caso del problema RC101. En este sentido, para aquellos problemas donde existen nodos con cierta flexibilidad de inserción como son los RC102, RC103, y RC104, nos encontramos con que los resultados arrojados por el modelo son similares al caso de los problemas *R1*, ya que el valor medio de la distancia incremental sobre las mejores soluciones encontradas, asciende a un 47%, cuando en el caso de los problemas *R1* era de un 45% aproximadamente. En este sentido, los resultados empeoran a medida que se incrementa el número de nodos con ventanas de tiempo estrechas.

Para la segunda mitad de los ejemplos de este tipo podemos observar algo muy interesante, que ya ocurría en los ejemplos *R1*. Nos referimos a la importancia de la existencia de ventanas de tiempo bilaterales, si bien no en su totalidad, como ocurre en el caso del RC101. Es necesario tener en cuenta que el algoritmo desarrollado en este trabajo se guía principalmente tanto para la selección de los nodos semilla como de los sucesivos nodos a ser insertados, por un criterio estrictamente temporal, de manera que se atiende en primer lugar a aquellos nodos más críticos, bien por sus holguras de tiempo, o bien por sus momentos de cierre. El hecho de que existan ventanas de tiempo estrechas implica que estos nodos siempre van a ser asignados en las mismas franjas horarias, se utilice el algoritmo que se utilice, de manera que implican una serie de decisiones fijas en la selección de los mismos, restringiendo enormemente la flexibilidad del problema. Por ello el algoritmo desarrollado en este trabajo es capaz de identificar este tipo de nodos y de programar las rutas, en base a este criterio. En este caso los problemas más restringidos como el RC105 se resuelven mejor que otros donde existe una mayor flexibilidad, como los RC106, RC107 y RC108. En estos otros problemas con mayor flexibilidad, generalmente son mejores otro tipo de algoritmos con búsquedas más intensivas.

2.4 Grupo de problemas RC2

Dentro de este grupo de problemas, la topología de los nodos se corresponde con la expuesta anteriormente para el caso de los problemas de tipo *RC1*, donde existe una dispersión aleatoria de algunos nodos, mientras que otros se mantienen agrupados en pequeños clusters. A diferencia de los problemas *RC1*, en este caso el horizonte de planificación, marcado por el intervalo de apertura del depósito, es más amplio, de manera que las rutas serán más largas, y por ello el número de vehículos a utilizar se reducirá. Esta longitud de las rutas también se favorece con la consideración de capacidades más amplias de carga para los vehículos, de manera que es posible que realicen un mayor número de servicios. Igualmente las restricciones temporales de los nodos no son tan exigentes como en el caso de los problemas tipo *1*.

Dentro de este grupo de problemas se han obtenido los siguientes resultados.

	R	*β*	*γ*	Distancia	NV	Mejor Resultado	Exceso en %
RC201	7	1,81	1,03	2198	17	1406,94	56%
RC202	6	1,62	1,00	1435	6	1389,57	3%
RC203	8	1,37	1,06	1392	8	1060,45	31%
RC204	7	1,57	1,01	1092	7	799,12	37%
RC205	11	1,57	1,05	1793	14	1302,42	38%
RC206	9	1,91	1,02	1669	12	1153,93	45%
RC207	10	1,56	1,00	1599	13	1062,05	51%
RC208	8	1,60	1,00	1402	8	829,69	69%
Acumulado	66	13	8	12580	85	9004	40%
Media	8	1,63	1,02	1573	11	1126	40%

Tabla 4.9 Resultados de los problemas tipo *RC2*

Número de rutas iniciales o nodos semilla

En el caso de este tipo de problemas generalmente el número de nodos semilla a considerar es bastante reducido debido a las características de los problemas tipo *2*. Sin embargo merece la pena destacar que aún siendo un valor bajo, es bastante más elevado de lo deseado ya que es posible resolver estos problemas con un número medio de vehículos cercano a 4, según los mejores resultados encontrados hasta la fecha. Durante la planificación se precisan un mayor número de vehículos resultando por término medio un número de rutas cercano a 11. En este sentido cabe destacar al igual que ocurre con los problemas *R2*, la necesidad de generar nuevas rutas durante la planificación motivado en parte por los desplazamientos excesivos que se han de realizar en algunas asignaciones de nodos, y que provocan la huida de los vehículos de aquellas zonas donde todavía existen nodos sin visitar, de manera que se necesitan nuevos vehículos para el servicio de los mismos, lo que ocurre con mayor frecuencia en los problemas donde existen mayores restricciones temporales, y que generalmente es donde se obtienen peores resultados, como es el caso del RC201, RC205, RC206, y RC207, donde los incrementos en el número de vehículos son mayores.

Parámetro *β*

En el caso del limitador de distancias para las inserciones simples nos encontramos con bastante elevados, lo que implica que es preferible aumentar las zonas de inserción para encontrar aquellos nodos para ser insertados entre un nodo origen y un nodo destino, de manera que se reduzcan el número de vehículos necesarios. Estos incrementos en las zonas de inserción también obedecen a la distribución de los nodos en el plano, y que en ocasiones están agrupados, de manera que para encontrar nodos a ser insertados es necesario recorrer amplias distancias fuera del cluster donde se halle el vehículo en ese momento. En este caso ocurre algo muy similar a lo que acontece con los problemas *R2*, y de ahí que los valores de β sean similares en ambos casos.

Parámetro γ

En el caso del multiplicador de zonas de inserción para inserciones dobles nos encontramos con que generalmente tiene escasa importancia, ya que obtiene valores cercanos siempre a la unidad. Esto implica que generalmente no se amplían las zonas de inserción para contemplar posibles inserciones conjuntas de dos nodos, lo cual obedece principalmente a dos motivos. En primer lugar, la zona de inserción ya es lo suficientemente grande como para realizar estas dobles inserciones, ya que el valor de β es elevado, y en segundo lugar, la distribución parcialmente clusterizada de los nodos implica que generalmente los nodos a ser insertados también se suelen encontrar bastante cerca entre sí, de manera que no es necesario ampliar estas zonas de inserción; algo que sí ocurría en los problemas tipo *R2*, y de ahí que los valores de γ en estos casos fueran ligeramente superiores.

Si atendemos a la composición de los problemas del grupo *RC2*, entonces podemos obtener alguna conclusión adicional.

	Nº Ventanas Bilaterales	Nº Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
RC201	100	0	120	-	10
RC202	75	25	120	Solamente tc	10
RC203	50	50	120	Solamente tc	10
RC204	25	75	120	Solamente tc	10
RC205	100	0	Entre 40 y 500	-	10
RC206	100	0	240	-	10
RC207	100	0	Entre 120 y 500	-	10
RC208	100	0	Entre 240 y 500	-	10

Tabla 4.10 Composición de los problemas *RC2*

En las líneas anteriores se comentaba la mejor bondad de cálculo del algoritmo en los problemas donde existía cierta flexibilidad temporal, como ocurrían en los grupos de problemas anteriores. En este caso se puede apreciar lo mismo, ya que es en los problemas RC202, RC203 y RC204 donde se obtienen los mejores resultados. En estos tres casos, la existencia de nodos no sujetos a ventanas bilaterales permite un buen funcionamiento del mecanismo de inserción, de manera que permite la utilización de un menor número de vehículos, reduciendo igualmente la distancia total recorrida. Por contra cuando el número de nodos sujetos a ventanas de tiempo bilaterales es del 100%, entonces el mecanismo de inserción no funciona con todo su potencial. Igualmente cuando el número de nodos libres aumenta, las soluciones del problema también aumenta, por lo que una vez más el algoritmo se muestra especialmente bueno cuando existe un cierto equilibrio entre nodos con ventanas bilaterales y nodos con ventanas unilaterales.

Dentro de este grupo llama especialmente la atención los resultados obtenidos para el problema RC203, donde la distancia total recorrida por la flota solamente es un 3% superior a la mejor solución obtenida hasta la fecha para este problema. Esta circunstancia no es casual, ya que se debe a las características internas del algoritmo

diseñado en este trabajo. Resultados similares se obtenían para los problemas R202, R204 y R207, donde existía una mezcla entre el número de nodos sujetos a ventanas bilaterales y el número de nodos con ventanas unilaterales. De esta manera los primeros permiten guiar las decisiones de asignación de los sucesivos nodos a las rutas, mientras que los segundos permiten un buen funcionamiento del mecanismo de inserción. Igualmente se da la circunstancia común de que las ventanas de tiempo son superiores a los tiempos de servicio, por lo que refuerza la posibilidad de realizar un mayor número de inserciones entre los nodos origen y destino en cada asignación, ya que las holguras resultantes son mayores, algo que no ocurre cuando las ventanas de tiempo son iguales a los tiempos de servicio, ya que dificultan las inserciones de nodos en el camino.

2.5 Grupo de problemas C1

Este grupo de problemas expuesto por Solomon se caracteriza por contemplar la distribución de los clientes en clusters o grupos homogéneos, aleatoriamente dispersos. En este sentido, los clientes se agrupan por zonas. La topología tradicional de este tipo de problemas se corresponde con la siguiente figura..

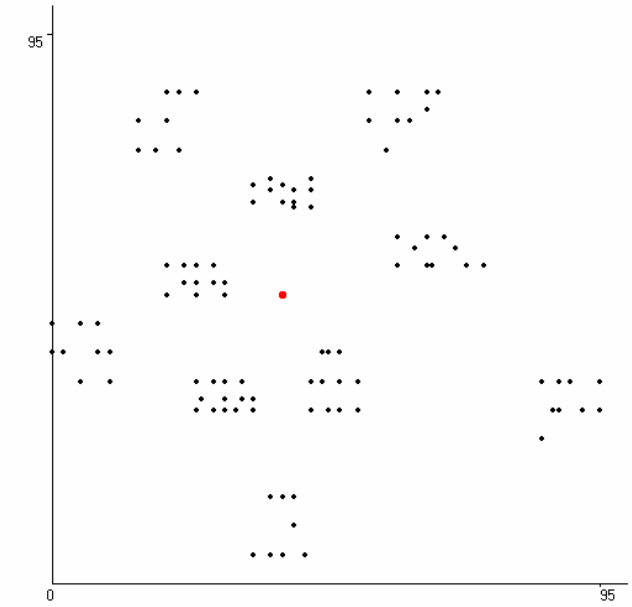


Figura 4.3 Topología clásica de los problemas tipo *C*

En el caso concreto de los problemas de tipo *C1*, el horizonte de planificación es reducido de manera que los vehículos en consideración realizarán pocas visitas, y por ello será necesario utilizar múltiples vehículos para realizar todas las visitas. Igualmente los clientes tienen fuertes restricciones temporales, marcadas por una estrecha ventana de tiempo para el depósito, y unas capacidades reducidas de los vehículos.

Una vez aplicado el modelo desarrollado a los ejemplos de este grupo de problemas, se han obtenido las siguientes soluciones, para los valores de los parámetros indicados en la siguiente tabla.

	R	*β*	*γ*	Distancia	NV	Mejor Resultado	Exceso en %
C101	10	1,00	1,00	2309	36	828,94	179%
C102	1	1,92	1,00	1351	17	828,94	63%
C103	8	1,09	1,00	1381	11	828,06	67%
C104	13	1,50	1,00	1317	14	824,78	60%
C105	10	1,00	1,00	2577	35	828,94	211%
C106	10	1,00	1,00	2544	35	828,94	207%
C107	10	1,00	1,00	2672	31	828,94	222%
C108	15	1,27	1,03	1819	25	828,94	119%
C109	18	1,05	1,00	1629	22	828,94	97%
Acumulado	95,00	10,83	9,03	17599	226	7455	136%
Media	10,56	1,20	1,00	1955	25	14082	136%

Tabla 4.11 Resultados de los problemas tipo *C1*

Número de ruta iniciales, o nodos semilla

Al igual que en los ejemplos de los grupos *R1* y *RC1*, el número de nodos semilla con los que se suele comenzar la planificación de este grupo de problemas suele ser bastante reducido en comparación con el número final de rutas. En general este valor del número de rutas iniciales para la planificación suele estar próximo a 10 u 11 rutas, correspondiéndose con el límite inferior de rutas para este tipo de problemas. Sin embargo, a medida que avanza la programación, este número se incrementa dado que no es posible atender a los sucesivos nodos por parte de las rutas iniciadas, y saturadas con visitas intermedias.

Parámetro *β*

En este grupo de problemas el comportamiento de β es bastante heterogéneo debido a la mezcla de tipos de problemas existentes. Por un lado nos encontramos con los problemas en los que no existe la posibilidad de insertar nodos en camino dadas las restricciones temporales de los nodos. Estos son el C101 y el C105, donde no se pueden realizar inserciones en el modelo propuesto. Por ello el valor de β es 1. En el caso de los problemas C102, C103 y C104, sí es posible y de hecho se insertan nodos intermedios dado que existe un conjunto de nodos donde no hay límite inferior en sus ventanas de tiempo, por lo que permanecen abiertos durante amplios períodos de tiempo, dispuestos a ser insertados. Lo mismo ocurre con los problemas del C106 al C109, ya que existe la posibilidad de realizar algunas inserciones entre los nodos seleccionados para ser asignados a las rutas, y las cabezas de ruta correspondiente. Sin embargo de entre estos problemas, solamente se permiten inserciones en los dos últimos ejemplos, ya que es en estos donde la amplitud de las ventanas de tiempo de los nodos en cuestión permiten la

realización de este tipo de inserciones, mientras que en el caso de los problemas C106 y C107, la estrechez de las mismas y la longitud de los desplazamientos necesarios para realizar estas inserciones impiden la ejecución de las mismas. No olvidemos que en este tipo de problemas los nodos están agrupados, y generalmente las inserciones serían de nodos que están localizados en otros clusters por lo que los desplazamientos serían muy amplios, provocando generalmente peores soluciones.

Parámetro γ

Los comentarios sobre el parámetro γ en este caso son obviados dada la escasa importancia que tiene el multiplicador de distancias en los ejemplos donde los nodos están agrupados en clusters. Por ello es necesario recordar que en este tipo de problemas las inserciones de nodos entre un nodo ya asignado a una ruta, y la cabeza de ruta correspondiente van a estar enormemente condicionadas por los excesivos desplazamientos que sería necesario realizar para seleccionar nodos que estuviesen abiertos, pero que al mismo tiempo estarían localizados en otros clusters. Es por esta razón que el algoritmo rechaza estas visitas a nodos lejanos optando por las esperas prolongadas.

Si atendemos al detalle de los problemas de este grupo, nos encontramos con los siguientes datos

	Nº Ventanas Bilaterales	Nº de Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
C101	100	0	Variado, < ts	-	90
C102	75	25	Variado, < ts	Solamente tc	90
C103	50	50	Variado, < ts	Solamente tc	90
C104	25	75	Variado, < ts	Solamente tc	90
C105	100	0	Variado, < ts	-	90
C106	100	0	Entre 90 y 150	-	90
C107	100	0	180	-	90
C108	100	0	Entre 150 y 300	-	90
C109	100	0	360	-	90

Tabla 4.12 Composición de los problemas *C1*

En este caso también merece la pena establecer algunos comentarios sobre los resultados del algoritmo en este grupo de ejemplos. En primer lugar, y al igual que viene ocurriendo en los anteriores grupos de ejemplos, el comportamiento del algoritmo en el caso de los problemas donde el 100% de los nodos tienen ventanas de tiempo estrechas (i.e. C101), el comportamiento del modelo desarrollado es bastante malo, debido a que los mecanismos de inserción no funcionan no siendo posible insertar nodos. En este sentido las mejoras en los resultados vendrían motivadas por la elección entre esperar por parte de las rutas más cercanas a los nodos críticos en cuestión, o realizar grandes recorridos por parte de rutas que se encuentren más lejanas, algo que directamente no contempla este algoritmo, ya que siempre se selecciona la ruta a la que está más cercano un nodo en estudio, independientemente de si ha de realizar una espera grande o no.

En segundo lugar, y también como ocurría en el caso de los anteriores ejemplos, cuando el número de nodos con ventanas reducidas no es muy elevado (desde 0 hasta el 75% de los nodos totales), y convive con algunos nodos libres, o donde no se aplican estas ventanas de tiempo tan restringidas, como es el caso de los problemas C102, C103, y C104, entonces el comportamiento del algoritmo mejora, obteniendo distancias incrementales medias, sobre las mejores soluciones de entorno al 63%. Si bien en este caso los resultados son peores que en el caso de problemas *R1*, donde esta distancia era del 45%, o en el caso de los problemas *RC1* donde esta distancia era del 47%. Esta observación también es lógica dado que la distribución de los nodos en conjuntos implica mayores recorridos para realizar posibles inserciones entre las cabezas de rutas y los nodos en estudio, por lo que a un mayor agrupamiento de los nodos, se obtienen peores soluciones provocadas por desplazamientos inútiles.

También es observable que las inserciones solamente se producen en aquellos casos donde existen estos nodos libres para ser insertados, y que se corresponden con los problemas C102, C103, C104, C108 y C109. En este sentido, los parámetros β y γ solamente obtienen valores mayores que uno en estos ejemplos, mientras que en los demás ejemplos, no tienen sentido el mecanismo de inserción contemplado en el modelo, dada la inexistencia de nodos a ser insertados entre cada nodo seleccionado y las cabezas de rutas.

2.6 Grupo de problemas C2

Dentro del grupo de problemas *C2* nos encontramos con la topología comentada para el caso de los problemas clusterizados de tipo *C1*, donde los clientes se encuentran agrupados por zonas, pero en este caso el horizonte de planificación vuelve a ser muy amplio, de manera que las rutas serán largas, y contarán con un elevado número de visitas, a lo que contribuye una amplia capacidad de carga de los vehículos.

	R	***β***	***γ***	**Distancia**	**NV**	**Mejor Resultado**	**Exceso en %**
C201	16	1,00	1,00	2106	25	591,56	256%
C202	4	1,93	1,00	731	4	591,56	24%
C203	4	1,79	1,00	871	4	591,17	47%
C204	5	1,05	1,00	1102	7	590,60	87%
C205	18	1,93	1,07	1877	19	588,88	219%
C206	16	1,93	1,05	1819	19	588,49	209%
C207	17	1,35	1,00	1741	17	588,29	196%
C208	14	1,20	1,00	1635	14	588,32	178%
Acumulado	94,00	12,18	8,12	11882	109	4719	152%
Media	11,75	1,52	1,02	1485	14	590	152%

Tabla 4.13 Resultados de los problemas tipo *C2*

Número de rutas iniciales, o nodos semilla

En este conjunto de problemas el número de nodos semilla o rutas iniciales de partida debería de ser bastante reducido, dado que generalmente son problemas que se resuelven con pocos vehículos. Salvo para los problemas C202, C203 y C204, en el resto nos encontramos con que el número de nodos es elevado. Esta circunstancia deriva de las bajas restricciones de este conjunto de problemas, y al hecho de que las ventanas de tiempo que afectan a la mayoría de los nodos están bastante lejanas en el tiempo. Esto provoca que la selección de nodos se realice en función de sus momentos de cierre, por lo que se escogerán momentos de cierre más bajos, pero aún así serán muy elevados. Una vez seleccionados estos nodos, solamente se insertarán hasta un máximo de dos nodos intermedios, por lo que se perderá un potencial muy grande de realizar un mayor número de inserciones, provocando que los nodos no asignados a estas rutas iniciales tengan que ser visitados por otras nuevas rutas, de manera que se incrementa durante la planificación el número de vehículos necesarios para realizar las visitas.

Quizás en estos casos donde las ventanas de tiempo son tan amplias, y por ello las restricciones temporales son tan reducidas, hubiera sido preferible seleccionar los nodos por sus momentos de apertura, y no por sus momentos de cierre. Igualmente el contemplar la posibilidad de inserción de más nodos aportaría grandes mejoras a las soluciones.

En el caso de los problemas C202, C203 y C204, el número de nodos semilla es reducido, y además se da la circunstancia al igual que en el resto de problemas de tipo *2*, que este valor de las rutas iniciales, suele ser coincidente con el valor de las rutas finales, de manera que no se incrementan las necesidades de vehículos durante la programación. Para estos tres ejemplos se da una coincidencia de nodos fuertemente restringidos, con nodos libres para ser insertados, de manera que en estos tres casos en concreto, sí es posible realizar inserciones de nodos en camino de forma eficiente, tal y como se refleja en los valores de los parámetros β y γ.

Parámetro β

En este conjunto de ejemplos, el parámetro β también tiene una importancia destacada derivada de la amplitud del horizonte de planificación, y consecuentemente la amplitud de los intervalos entre ventanas de tiempo, y la amplitud de las propias ventanas. De esta manera es destacable el buen funcionamiento del mecanismo de inserción, que selecciona aquellos nodos intermedios. Igualmente, y tal y como ocurría en los casos anteriores, solamente muestra valores mayores a 1 en aquellos casos donde existen nodos posibles para ser insertados, que en este conjunto de problemas se corresponde con todos los ejemplos, salvo en el primero, donde las restricciones temporales impiden la inserción de ningún nodo.

Además es el conjunto para el que el valor de β es por término medio mayor, lo que también resulta lógico si pensamos en el tipo de problemas con el que nos encontramos.

Para ellos es necesario realizar grandes desplazamientos para encontrar nodos cuya inserción sea posible dado que se encuentran agrupados, por lo que es necesario ampliar la zona de inserción para encontrar estos nodos a ser candidatos. De ahí el elevado valor de β.

Parámetro γ

En este caso, y dada la escasa importancia de γ tampoco se realizan comentarios destacables sobre el mismo, dado que la zona de inserción ya viene marcada por los elevados valores de β, de manera que no es necesario multiplicar esta zona para ampliarla.

Si atendemos a las particularidades de cada uno de los ejemplos de este grupo, entonces podemos obtener alguna información adicional.

	Nº Ventanas Bilaterales	Nº de Ventanas Unilaterales	Tamaño Ventanas Bilaterales	Tamaño Ventanas Unilaterales	Tiempo de servicio
C201	100	0	160	-	90
C202	75	25	160	Solamente tc	90
C203	50	50	160	Solamente tc	90
C204	25	75	160	Solamente tc	90
C205	100	0	320	-	90
C206	100	0	Entre 320 y 640	-	90
C207	100	0	Entre 200 y 1000	-	90
C208	100	0	640	-	90

Tabla 4.14 Composición de los problemas *C2*

Lo primero que merece la pena comentar es que los resultados obtenidos para este conjunto de problemas son los peores de entre todos los grupos descritos, a pesar de ser también los problemas más sencillos de resolver dadas sus pocas restricciones impuestas. En este sentido se plantea una paradoja importante, que deriva de una serie de razones que se comentan a continuación.

El algoritmo se guía al principio por las holguras derivadas de las ventanas de tiempo de los nodos más críticos, de manera que estas holguras marcan el número de nodos semilla a ser considerados. Si las holguras son muy amplias, y aún así solamente se contemplan un máximo de dos inserciones en el camino, entonces se registrarán grandes esperas. Por esto el modelo obtiene su mayor potencial de funcionamiento en circunstancias donde las restricciones temporales son más fuertes, pero sin llegar a alcanzar al 100% de los nodos, tal y como se comentó en los casos anteriores, ya que limitan las posibilidades de mejora a través del mecanismo de inserción de nodos.

A pesar de ello el algoritmo obtiene buenos resultados para los problemas C202, C203 y C204, con un valor medio de la distancia incremental sobre los mejores resultados que asciende al 52% aproximadamente. En este sentido está en la media alcanzada para los

casos donde las ventanas de tiempo son fijas, ya que además existen nodos para realizar inserciones intermedias.

En general este tipo de problemas se resuelve generalmente bien por otros métodos que se fundamenten en los mecanismos de cluster primero y rutas después.

2.7 Resumen y Conclusiones

En este apartado se establece un resumen sobre los resultados obtenidos en la fase de experimentación con cada uno de los bloques de problemas tipo desarrollados por Solomon (1987), tal y como se recoge en los apartados anteriores. Para ello se ha elaborado la siguiente tabla con los datos medios más característicos sobre los resultados obtenidos, diferenciando cada uno de los grupos en consideración.

	Parámetros y Resultados	Clase *1*	Clase *2*
R	*R*	9,58	6,45
	β	1,20	1,70
	γ	1,03	1,07
	Número de Rutas	26	8
	Distancia	1955	1239
	Mejor Distancia	1211	960
	Incremento medio	60%	29%
RC	*R*	9,3	8
	β	1,49	1,63
	γ	1,10	1,02
	Número de Rutas	21	11
	Distancia	2247	1573
	Mejor Distancia	1385	1126
	Incremento medio	62%	40%
C	*R*	10,5	11,75
	β	1,20	1,52
	γ	1	1,02
	Número de Rutas	25	14
	Distancia	1955	1485
	Mejor Distancia	828	590
	Incremento medio	136%	152%

Tabla 4.15 Valores medios de los parámetros y resultados en los problemas de Solomon (1987)

Lo primero que cabe destacar son los resultados obtenidos para los grupos de problemas tipo *C*, donde los nodos se encuentran agrupados en zonas o clusters. Para estos ejemplos generalmente el algoritmo proporciona malos resultados, a pesar de ser los problemas más fáciles de resolver, y que generalmente permiten la obtención del óptimo en la mayor parte de los algoritmos desarrollados por los autores en la literatura. En este sentido es necesario destacar que el algoritmo aplicado y desarrollado en este trabajo se ideó y fundamentó en la problemática de nodos dispersos y fuertemente restringidos por ventanas temporales, de manera que se optó por un criterio principalmente temporal tanto para la selección de nodos semilla (aquellos con menor flexibilidad de asignación), como

para las sucesivas selecciones de nodos críticos a ser añadidos a las rutas ya iniciadas, (seleccionados según los tiempos de cierre más tempranos). Por ello en este conjunto de problemas, son más adecuados los algoritmos que combinan los criterios temporales, con criterios de decisión fundamentados en las distancias espaciales, y de ahí el origen de estos malos resultados. A pesar de ello, en el caso de los problemas de tipo *C2*, generalmente se obtienen mejores resultados que en el caso de los problemas de tipo *C1*, donde las restricciones temporales del horizonte de planificación son superiores, imposibilitando generalmente la aplicación del mecanismo de inserción propuesto en este trabajo.

Para el resto de problemas, nos encontramos con que generalmente los clientes se encuentran más o menos dispersos en el plano, ya sea de manera uniforme, como en el caso de los problemas de tipo *R*, como combinados en grupos o zonas, como es el caso de los problemas de tipo *RC*.

Dentro de estos conjuntos de clientes, el algoritmo mejora sustancialmente cuando la dispersión de los mismos se incrementa, de manera que dejan de estar completamente clusterizados como ocurría en los de tipo *C*. En este sentido se aporta una mayor flexibilidad al funcionamiento del mecanismo de inserción, ya que generalmente entre un nodo crítico a ser añadido a una ruta preestablecida, y el nodo cabeza de ruta, existen otros nodos que se encuentran disponibles para ser insertados, de manera que los tiempos de espera se saturan con visitas intermedias. Sin embargo, a pesar de la existencia física de estos nodos, a veces su visita está condicionada por las ventanas temporales existentes. En este sentido si el 100% de los nodos están sujetos a ventanas de tiempo estrechas, cercanas a los tiempos de servicio, entonces los resultados del algoritmo empeoran. Caso contrario, si aún existiendo estas ventanas de tiempo estrechas, incluso equivalentes a los tiempos de servicio, también existen otros nodos donde las restricciones temporales no son tan rígidas, entonces los resultados del algoritmo también mejoran sustancialmente. Esto es debido a los fundamentos del algoritmo, ya que para un buen funcionamiento requiere la existencia de estos nodos críticos para que guíen la selección de los nodos que han de ser sucesivamente añadidos a las rutas en consideración. Los resultados empeoran sensiblemente cuando se incrementa el tamaño de las ventanas de tiempo de los nodos, lo que implica una mayor relajación de las restricciones del problema. En este sentido la conclusión no es que el algoritmo se comporte mal, sino que los demás algoritmos que generalmente aplican criterios de búsqueda intensivos tienen una mayor flexibilidad para encontrar los valores cercanos al óptimo, arrojando buenas soluciones.

Más en concreto, generalmente se obtienen mejores soluciones en los problemas donde las rutas son más largas, que en aquellos problemas donde existen espacios cortos de planificación. Es decir que el algoritmo funciona mejor para los casos donde el número de nodos existentes es muy elevado en comparación con el número de vehículos disponibles, o lo que es lo mismo, la necesidad de realizar un elevado número de repartos con pequeñas flotas de vehículos. Este tipo de problemas se corresponde con los ejemplos de tipo *2*, donde las rutas generalmente cuentan con un número de nodos que varía entre 15 y 30 nodos, mientras que en el caso de los problemas de tipo *1*, las rutas son mucho más

cortas, ya que el número de nodos oscila entre 5 y 10 nodos, siendo necesario un mayor número de vehículos para realizar todas las visitas. Esto es debido a que a medida que se aumenta el horizonte temporal, las posibilidades de inserción también se incrementan, siempre y cuando existan nodos disponibles, tal y como se comentaba en el párrafo anterior.

En concreto los problemas que mejor se resuelve por parte del algoritmo en cuestión son los del grupo *R2* y *RC2*, donde la dispersión geográfica de los clientes es similar (salvando las diferencias comentadas en cada caso), y además existen los suficientes nodos libres para ser insertados como para que el mecanismo de inserción propuesto arroje buenos resultados. En concreto, cuando existen estos nodos generalmente los resultados obtenidos por el modelo en cuestión solamente distan en un 15% de las mejores soluciones obtenidas hasta el momento por el resto de los algoritmos estudiados, siendo el problema *RC202* el que proporciona los mejores resultados de entre todos los problemas resueltos, con solamente un 3% de incremento sobre la distancia menor encontrada hasta la fecha, algo inaudito para un algoritmo de construcción de rutas, tan rápido como es el desarrollado y propuesto en este trabajo.

La conclusión general sobre el algoritmo, es que se trata de un modelo que funciona mejor en condiciones muy adversas, de manera que los resultados empeoran a medida que se reducen las restricciones del problema en cuestión. En este caso los resultados aportados por otros métodos suelen estar muy cercanas al óptimo, ya que disponen de una mayor flexibilidad para probar y buscar nuevas soluciones en el espacio de soluciones del problema. En este sentido las menores restricciones del problema, especialmente las de tipo temporal perjudican la funcionalidad de este algoritmo, lo cual es bastante intuitivo si recordamos que el algoritmo se sirve de estas restricciones (principalmente las de tipo temporal), para construir las rutas de la solución.

Al final de este capítulo se comentan alguna características que el algoritmo propuesto no contempla, pero cuya inclusión, una vez modelizada y programada, proporcionaría mejoras sustanciales en los resultados.

Si atendemos a las conclusiones obtenidas sobre los parámetros, debemos de resaltar las siguientes afirmaciones.

En primer lugar, el valor de los nodos semilla generalmente aporta una gran variabilidad a los resultados arrojados por el algoritmo, sobre todo en aquellos problemas donde el horizonte de planificación es más breve, lo que se corresponde con los problemas de tipo *1*. En este sentido la implementación de múltiples combinaciones con este parámetro es fundamental para obtener el valor mejor de los nodos semilla a considerar, y que al mismo tiempo suele ser bastante heterogéneo entre los diversos problemas resueltos. Además es necesario considerar que generalmente es preferible comenzar con un número bastante reducido de nodos semilla en comparación con el valor final del número de rutas de la solución. Esto se debe a que generalmente se crean una cantidad de rutas durante la planificación que suele equivaler al 100% de las rutas inicialmente creadas, de manera

que es preferible comenzar con pocos vehículos y esperar a que nuevos vehículos se requieran durante la planificación. En este sentido generalmente las últimas rutas creadas suelen contener muy pocos nodos, siendo rutas malas, susceptibles de ser mejoradas. En este sentido también es posible mejorar la funcionalidad del algoritmo a través de la inclusión de nuevos criterios de valoración en la creación de nuevas rutas, tal y como se recoge al final de este capítulo.

Sin embargo en el caso de los problemas de tipo *2*, que se resuelven con pocas rutas, generalmente es preferible comenzar con un número de vehículos proporcionalmente mayor que en el caso de los problemas del tipo *1*, pero que en todo caso es muy reducido, ya que oscila entre 4 y 6 rutas. A diferencia que en caso anterior, generalmente no se suelen crean nuevas rutas durante la planificación, que generalmente termina con el mismo número de rutas con las que se comienza. En este sentido el algoritmo se comporta mejor que en los casos donde el horizonte de planificación es más restringido.

Con respecto al parámetro β, este ha mostrado todos su potencial, en todos los problemas donde la inserción de nodos entre las cabezas de ruta, y los nodos críticos a ser añadidos a las rutas inicializadas era posible. En este sentido nos encontramos con que según la tipología del problema en cuestión, los valores de β variaban enormemente, tal y como se muestra en el cuadro esquemático presentado anteriormente. De todos modos, en raras ocasiones este valor de β es superior a *2*, lo que implica que por regla general se rechazan los nodos a ser insertados que supongan incrementos superiores al 100% de la distancia entre la cabeza de ruta, y el nodo crítico a ser añadido a esa ruta. En este sentido se suele optar por las esperas, caso de no existir estas posibles inserciones. Los valores de β indican comportamientos diferentes según la tipología de problemas, de manera que en ocasiones es preferible ser más restrictivo en estas zonas de inserción (generalmente en los problemas de tipo *1*, donde existe un mayor número de rutas para visitar estos nodos), y en otras se suele ser más permisivo (caso de los problemas de tipo *2*, donde el número de rutas es mucho más reducido).

También es destacable el comportamiento del parámetro γ, que generalmente aporta poco valor en las soluciones de los problemas en cuestión, salvo cuando la importancia de los mecanismos de inserción es máxima, lo que se corresponde con los problemas de tipo *2*. Es en general en este tipo de problemas cuando el parámetro γ adquiere un cierto grado de protagonismo provocando ampliaciones de las zonas de inserción para el caso de las dobles inserciones, sobre todo en los problemas de tipo *RC*, donde en ocasiones los nodos a ser insertados están algo distantes por formar parte de pequeños clusters, tanto en el caos de los problemas de tipo *1* como en los problemas de tipo *2*. Principalmente este protagonismo se revela una vez más en los problemas de tipo *2*, ya que como se indicaba anteriormente, el número de rutas es menor, y por ello las posibilidades de vista de estos nodos por rutas alternativas, de manera que es preferible insertarlos a toda costa entre las rutas inicializadas y los nodos que sucesivamente se van seleccionando como nodos críticos.

En resumen, debido a la gran diversidad de problemas recogidos en los desarrollos de Solomon, hablar de un comportamiento global de la sensibilidad de estos parámetros resulta bastante irreal, dado que su comportamiento depende enormemente de las características intrínsecas de dichos problemas, y que no solamente se resumen en una clase (*R*, *RC* y *C*), o en un tipo (*1* y *2*), sino que también influye la proporción de ventanas unilaterales en contra de ventanas bilaterales, los tamaños de las mismas, los tiempos de servicio, etc. Por esto cada problema tiene sus particularidades, tal y como se comentó en este trabajo, de manera que en general, el comportamiento del algoritmo en cuestión es altamente heterogéneo según el problema en cuestión.

A pesar de ello, se han realizado diversos análisis del comportamiento de estos parámetros para la obtención de las soluciones de algunos casos en particular. Este análisis se ha concretado en contrastar y validar los resultados arrojados por el algoritmo para cada problema, atendiendo a las diferentes combinaciones posibles de todos los parámetros.

A continuación se recoge un ejemplo del análisis de sensibilidad realizado para uno de estos problemas, en concreto el R103.

2.7.1 Análisis de sensibilidad de parámetros: El ejemplo del R103

Para realizar este análisis de sensibilidad se computaron las diferentes soluciones obtenidas a través de las distintas combinaciones de parámetros. Teniendo en cuenta que se trata de un problema del tipo *R* y clase *1*, los parámetros que mayor incidencia van a tener sobre el cómputo de la solución final, van a ser el número de rutas iniciales o nodos semilla, y el valor del parámetro β. Por ello el análisis comienza con un estudio de las diferentes combinaciones de estos dos parámetros. Para evitar excesivos cálculos, y apoyándonos en la hipótesis de que en el espacio de soluciones, las buenas soluciones tienden a estar agrupadas, se ha simplificado la búsqueda a combinaciones del parámetro *R*: Número de nodos semilla, entre 5 y 24 (representadas por los valores desde C1 hasta C20 en el eje *x*), y el parámetro β, entre 1, y 2,5 (representado por las marcas de 1 a 25 del eje *y*) contemplando incrementos decimales del mismo. Para ello se han obtenido las 20 x 31= 620 soluciones posibles resultantes de las posibles combinaciones de parámetros. Estas soluciones se recogen en el anexo III de este trabajo, y se muestran en la figura 4.4.

En este punto recordamos también que la mejor solución obtenida por el algoritmo para este problema suponía unos valores de R=20, β=1.74 y γ=1.06

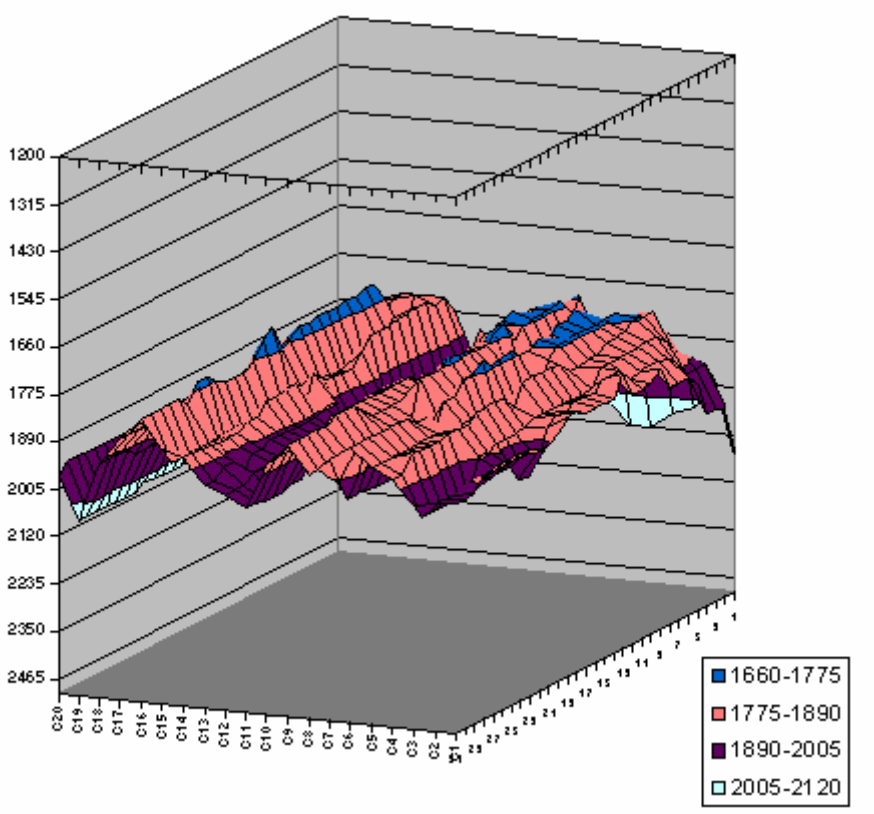


Figura 4.4 Espacio de soluciones para combinaciones de *R* y *β* en el problema R103

En este caso se han representado los valores de la distancia obtenida para las diferentes combinaciones del parámetro *R*, (representado en el eje *x*) con el parámetro *β* (representado en el eje *y*). Se ha realizado una inversión de los ejes para obtener una mejor visualización de los valores mínimos de la distancia (correspondientes a los puntos más elevados marcados en azul). En este sentido se identifican tres zonas en las que se hallan estos valores mínimos para la distancia total recorrida, de manera que requieren un mayor análisis para identificar exactamente la combinación para la que se encuentra la combinación mínima.

Si bien lo interesante de esta gráfica es la percepción de la variabilidad de los resultados ante las diferentes combinaciones de estos dos parámetros, lo que traducido a curvas de nivel se puede mostrar en la figura 4.5.

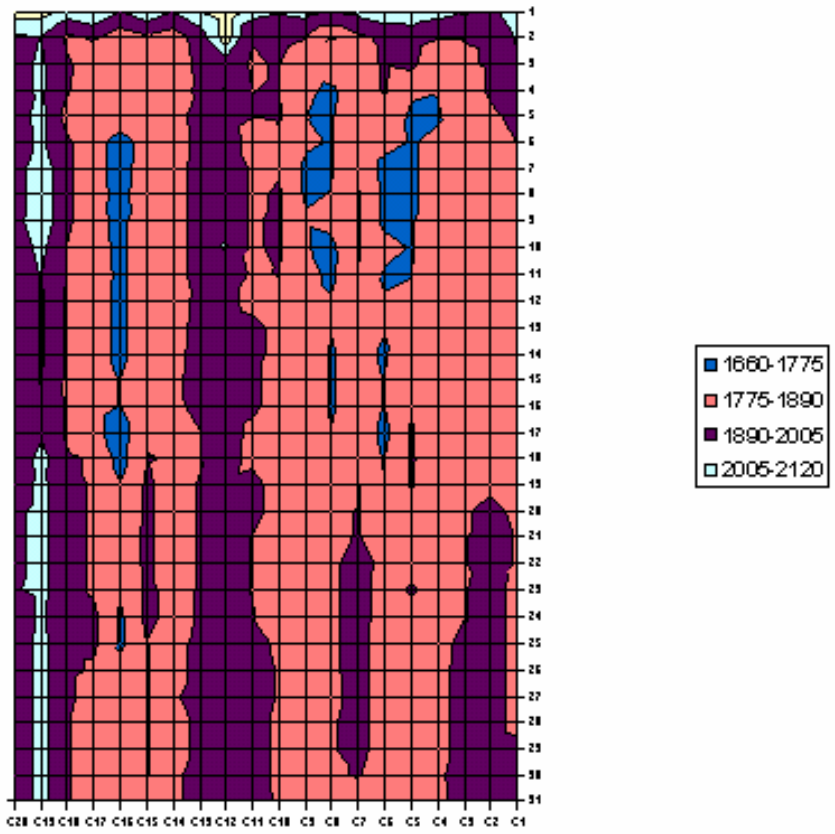


Figura 4.5 Curvas de nivel para el espacio de soluciones del problema R103

Las zonas marcadas en azul se corresponden con los valores más bajos de las distancias obtenidas a través de las posibles combinaciones. En este gráfico se percibe claramente donde se encuentran esos valores mínimos para la distancia total, que se corresponde con un valor de R de 20, y unos valores de β de entre 1.7 y 1.9. Por ello es conveniente explorar con mayor detenimiento esta zona y ver que ocurre con el parámetro γ, una vez que ya ha sido fijado el parámetro R en 20 rutas iniciales, y que resulta en la siguiente gráfica, representativa del corte longitudinal de la figura 4.4 para R=20:

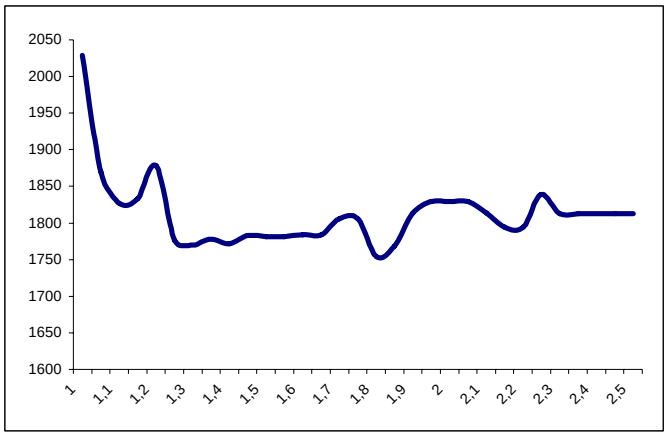


Figura 4.6 Perfil de soluciones para *R*=20 según los valores de β en el problema R103

Si exploramos el espacio de las soluciones correspondientes al pico mostrado en la figura anterior, y para el que se encuentran los valores mínimos de las soluciones arrojadas por el algoritmo, entonces es posible establecer las combinaciones para las que se obtienen esos valores mínimos. Estas combinaciones de los parámetros β y γ, y para un número de rutas iniciales correspondiente a R=20, se muestra en la figura 4.7 donde se representan los valores de γ entre 1.00 y 1.20 con incrementos centesimales, e ilustrados en el eje con los valores de 1 a 20. Igualmente se consideran los valores para β, entre 1.70 y 1.88 (representado en la gráfica en el eje horizontal), con incrementos centesimales, y representados en la gráfica como C1, hasta C19.

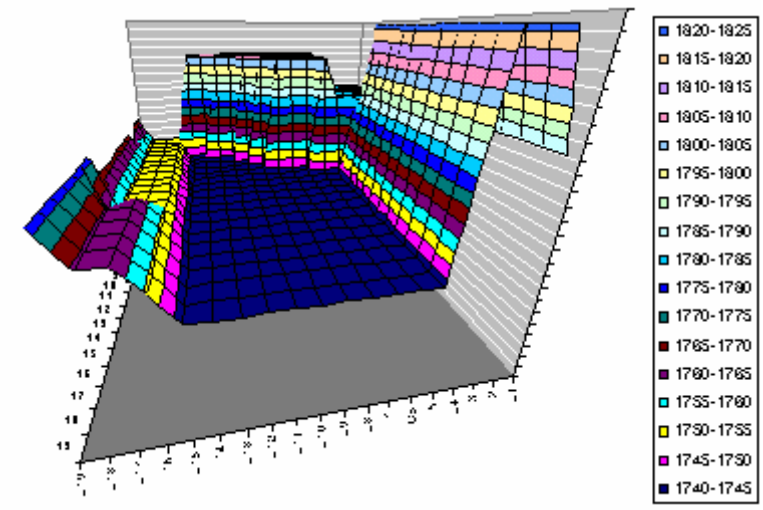


Figura 4.7 Espacio de soluciones para *R*=20, según los valores de β y γ

De esta manera y a través de la adopción de un enfoque más microscópico se obtienen todas las posibilidades de combinación entre β y γ, para las que se reportan los valores mínimos de las soluciones encontradas por el algoritmo, y que en la gráfica anterior se

corresponden con todos los puntos correspondientes al suelo representado en color azul oscuro. Los valores para la construcción de esta gráfica se recogen en el anexo III de este trabajo.

En este sentido el método se puede clasificar entre los métodos denominados de *Steepest Descent*, o de descenso más vertical, dado que es capaz de obtener la menor de todas las soluciones posibles para un intervalo de valores de los parámetros. Sin embargo no deja de ser un óptimo ficticio, ya que si bien los valores de los parámetros sí resultan ser óptimos, el método en sí no es capaz de encontrar la solución óptima del problema, ya que contempla determinadas imperfecciones, algunas de ellas susceptibles de corrección, tal y como se apunta al finalizar este capítulo

Con este sencillo ejemplo se ha pretendido ilustrar la eficacia de ajuste de los parámetros contemplados en el algoritmo, así como su operativa y su sensibilidad, que tal y como se apuntó anteriormente, dependen en gran medida del tipo de problemas con el que trabajemos, ya que para algunos casos sí se muestran eficaces algunos parámetros, y para otros casos, no.

En resumen, es posible afirmar que el método en cuestión es capaz de resolver todos los problemas presentados por Solomon, con mayor o menor éxito, y siempre a través del ajuste paramétrico recogido en el modelo. Para ello es posible implementar gran número de combinaciones que resultan en otras tantas posibles soluciones.

Por último cabe destacar la capacidad del algoritmo para generar un elevado número de soluciones, todas ellas factibles, en la medida en que cumplen con las restricciones del problema, salvo en el número de rutas, ya que se ha optado por no considerar su limitación. En este sentido puede integrarse dentro de los métodos heurísticos de construcción de rutas en paralelo, y fundamentado en los principios de los mecanismos de inserción, pero con las particularidades recogidas en el capítulo 3. En el apartado siguiente se establece una comparativa del método recogido en este trabajo con otros modelos analizados en la revisión bibliográfica, con la finalidad de establecer una valoración sobre la bondad y validez del método aquí planteado, y discutir en la última parte de este capítulo su aplicabilidad.

3 Comparativa con otros métodos

En el apartado anterior se experimentaba con la aplicación del algoritmo desarrollado a los problemas tipo presentados por Solomon, con la intención de analizar su comportamiento individualizado en cada uno de los casos desarrollados, así como un análisis puntual del comportamiento de los parámetros para cada uno de los grupos de problemas, y un análisis detallado de las características de los mismos. En este apartado se presenta a modo de discusión y comparativa, los resultados arrojados por los métodos presentados en el capítulo 2 para la resolución del VRPTW. Para ello se recogen los resultados resumidos de cada uno de los métodos en la resolución de la batería de problemas presentados por Solomon, para identificar cuáles han sido las evoluciones más notorias en los últimos tiempo, y discutir las aportaciones de este modelo en la línea de investigación aquí contemplada.

Cabe destacar el importante papel que juegan los métodos de construcción de rutas en la posterior postoptimización realizada por los métodos metaheurísticos, de manera que sin los primeros no sería posible resolver ningún tipo de problemas. También cabe destacar el poco trabajo existente sobre los algoritmos puramente de construcción de rutas, ya que la gran mayoría de los autores se fundamentan en el algoritmo *I1* de Solomon (1987), para obtener las soluciones iniciales. En este sentido el modelo presentado en este trabajo pretende profundizar, comprender, y analizar las reglas que guían los procedimientos de inserción de nodos para la construcción de rutas de transporte.

3.1 Resultados de los algoritmos más conocidos en los problemas de Solomon (1987)

En este apartado se presentan los resultados proporcionados por los diferentes autores de los algoritmos descritos en el capítulo 2, en la resolución de la problemática del problema del VRPTW.

Las siguientes tablas recogen los resultados arrojados por los diferentes autores contemplados en el capítulo 2, tal y como se reporta en los trabajos de los mismos. Por simplificar se han truncado los decimales en las soluciones relativas a la distancia total recorrida, ya que los datos incluidos en las tablas solamente se ofrecen con una finalidad meramente indicativa de las evoluciones sufridas por los mismos. Igualmente también se ha truncado debido a las diferencias mostradas por algunos autores en relación con las soluciones obtenidas por algunos de sus colegas, ya que en ocasiones no coinciden los datos del autor original con los datos recogidos por otros autores. Por esto algunos de los datos aquí recogidos pueden no corresponderse exactamente con las cifras aportadas por los autores originales.

Se han incluido como datos de la solución los valores medios obtenidos para el número de vehículos utilizados en las diferentes soluciones obtenidas, así como la distancia media para cada grupo de problemas, e igualmente se han incluido los datos relativos a las

soluciones obtenidas con el algoritmo presentado en este trabajo, y que en el apartado anterior se comparaban con las mejores soluciones obtenidas hasta el momento por parte de los mejores desarrollos.

En este sentido se incluyen para cada autor los mejores resultados obtenidos, y no los mejores globales, que en todo caso sería una comparación injusta.

3.1.1 Métodos de Construcción de rutas.

Realmente es en este grupo donde se ubica el algoritmo desarrollado durante el curso de la investigación realizada. En este sentido se ha obtenido un método de construcción de rutas en paralelo y de forma secuencial, a través de una saturación de visitas intermedias a medida que se avanza en el horizonte de planificación, lo que permite una aplicabilidad máxima a situaciones donde se requiera planificación dinámica de las rutas, tal y como se comentará al finalizar este capítulo.

Dentro de los métodos de construcción de rutas, nos encontramos con los trabajos de Solomon sobre los diferentes mecanismos de inserción propuestos para la resolución de los problemas del VRPTW. Igualmente se ha incluido en este grupo las soluciones obtenidas por Potvin et al. (1993) con su versión paralelizada del algoritmo de Solomon, y el algoritmo de Ioannis (2001). Estos serían los tres ejemplos que formarían el grupo de los algoritmos considerados puramente como de construcción de rutas, lo que resalta el escaso trabajo realizado en esta línea, no por la calidad de los mismos, sino por la escasa investigación realizada. Generalmente se asume la hipótesis de que este tipo de métodos son poco eficaces por naturaleza, ya que constan de múltiples inconvenientes, tal y como se ha reseñado en el apartado anterior, sin embargo pocos reconocen sus grandes ventajas, derivadas de resultar métodos deterministas, y por ello capaces de replicar cualquier solución, lo que no es posible en métodos estocásticos. Igualmente resultan ser los métodos más rápidos en el cómputo de las soluciones, y por último y quizás más importante, es que sirven de base para los segundos, de manera que a mayores velocidades de cálculo, y mejores resultado obtenidos, también se mejoran los resultados de los métodos metaheurísticos.

Construcción	***R1***		***R2***		***RC1***		***RC2***		***C1***		***C2***	
	Dist	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**
Solomon (1987)	1437	13,60	1402	3,30	1597	13,50	1682	3,9	951	10	692	3,13
Potvin y Rousseau (1993)	1509	13,30	1387	3,10	1724	13,40	1651	3,6	1343	10,67	797	3,38
Ioannu et al. (2001)	1370	12,67	1310	3,09	1512	12,50	1483	3,5	865	10	662	3,13
Guillén (2002)	1955	26,00	**1239**	8,00	2247	21,00	**1573**	11	1955	25	1485	14

Tabla 4.16 Comparación de los métodos de construcción de rutas

En la tabla 4.16 se muestran los resultados resumen para los diferentes grupos de problemas desarrollados por Solomon (1987). Se resaltan en negrita los casos en los que el algoritmo propuesto supera alguno de los resultados de otros autores. En este sentido cabe destacar la mejoría que presenta el método desarrollado en este trabajo en el caso de

los grupos de problemas *R2* y *RC2*, que tal y como se comentaba en el apartado anterior, son los ejemplos en los que el algoritmo presentaba una mayor eficacia y eficiencia, sobre todo en los casos particulares donde existían nodos con ventanas unilaterales, de manera que se presentaba una mayor flexibilidad en el mecanismo de inserción. Si bien, estos mejores resultados se obtienen con un mayor número de vehículos, aunque recordemos que el objetivo en el presente modelo es la minimización de la distancia total recorrida.

La diferencia con respecto al caso de los modelos de Solomon y Potvin es que el algoritmo presentado en este trabajo realiza una saturación incremental de las rutas, ya que a medida que se va avanzando en el contador temporal del horizonte de planificación se va atendiendo a las diferentes necesidades impuestas por los clientes, de manera que las rutas se van saturando a medida que avanzan los vehículos, y además esto se realiza con varios vehículos simultáneamente, de forma que se lleva a cabo una construcción paralela de las rutas. Sin embargo Solomon, establece la programación de las visitas a posteriori, en el sentido de que el algoritmo escoge a ciegas el nodo más distante del depósito, y a partir de ahí satura hacia delante y hacia atrás los nodos que maximizan los ahorros generados por sus respectivas inserciones en la ruta creada, hasta que ya no es posible realizar más inserciones. Una vez creada esta primera ruta, se inicia la segunda, y así sucesivamente hasta haber insertado todos los nodos del problema. En este sentido el método de Solomon parte de una selección de nodos semilla que pueden no estar abiertos al comienzo de la planificación, de manera que no existe una correlación entre la urgencia que ese nodo exige para ser atendido o asignado, y el momento de su selección para formar parte de una ruta en la solución. Por ello este tipo de métodos son poco adecuados para situaciones donde se requiera una planificación dinámica, o bien exista algo de incertidumbre sobre la definitiva realización de los pedidos por parte de los clientes, ya que su asignación no se hace con una mentalidad de programación temporal lógica, sino a través de un método de optimización global, en el que la solución a cualquier posible contingencia o cambio afecta enormemente a la solución previamente obtenida, mientras que en el algoritmo aquí presentado, los cambios en nodos cuya necesidad de atención sea posterior al reloj de la programación, no provocan cambios importantes en la planificación de las rutas, ya que hasta ese momento no se habrán tenido en cuenta.

En resumen, podemos afirmar que el trabajo aquí presentado supone un nuevo método heurístico de construcción de rutas que mejora algunos aspectos de los métodos clásicos de construcción de rutas, que en su mayoría se fundamentan en los planteamientos de Solomon (1987). Sin embargo, la asignatura pendiente es la adquisición de una mayor robustez en el cálculo de la solución de cualquier tipo de problemas, ya que su eficacia es bastante desigual según que tipo de problemas esté resolviendo. Igualmente sería necesario ser más crítico en la generación de nuevas rutas durante la programación, ya que muchas de las rutas adicionales creadas durante la programación solamente visitan uno o pocos clientes, de manera que serían susceptibles de ser reasignados, insertando estos clientes en rutas más completas.

3.1.2 Métodos de Mejora y Metaheurísticos

Se presentan los métodos de mejora contemplados en el capítulo 2, entre los que destacan algunos de los autores que habían desarrollado métodos de construcción previamente (i.e. Potvin et al. 1993), pero que posteriormente ofrecen sus versiones mejoradas. En este sentido muchos de estos trabajos contemplan la aplicación de los conocidos métodos de intercambio 2-opt, 3-opt, Or-opt, de Cruce, de Intercambio, etc. combinados de diferentes maneras sobre las soluciones iniciales generalmente obtenidas a través de los métodos de construcción de rutas expresados anteriormente, y mayoritariamente utilizando el método de Solomon *I1*, cuyos resultados se muestran en la tabla anterior.

Si bien la comparativa entre estos métodos y los de construcción de rutas no tienen ningún sentido en cuanto a los comentarios sobre mejora de eficacia, sí se ha incluido en este trabajo con una doble intención. En primer lugar, para mostrar el potencial de mejora ofrecida por los sencillos métodos de mejora desarrollados en la literatura, de manera que se consiguen reducciones de entre un 10 y un 20% sobre los resultados arrojados por los métodos de construcción. Y en segundo lugar para mostrar las evoluciones típicas que siguen los autores en el desarrollo de sus líneas de investigación, y que implica que los primeros en aportar trabajos sobre la resolución del VRPTW, primero enfocaron sus trabajos en la construcción de rutas, para posteriormente aplicar métodos de mejora, y finalmente realizar introspecciones en el campo de la búsqueda tabú y mayoritariamente sobre los algoritmos genéticos, pero siempre con la base y pilar fundamental de los algoritmos de construcción. A pesar se recogen los resultados de estos métodos con fines meramente comparativos de las posibilidades de unos y otros.

Mejora	**R1**		**R2**		**RC1**		**RC2**		**C1**		**C2**	
	Dist	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**
Thompson y Psaraftis (1993)	1367	13,1	1299	3,1	1534	13	1672	3,7	916	10	644	3
Potvin et al. (1995)	1381	13,33	1293	3,27	1545	13,25	1595	3,88	902	10	653	3,13
Russell (1995)	1317	12,7	1167	2,9	1523	12,4	1398	3,4	930	10	681	3
Antes y Derigs (1995)	1386	12,8	1367	3,1	1546	12,5	1598	3,4	955	10	717	3
Prosser at al. (1996)	1242	13,5	977	4,09	1408	13,5	1111	5,13	843	10	607	3,13
Shaw (1997)	1206	12,4			1363	12,1			828	10		
Shaw (1998)	1198	12,5			1361	12,1			828	10		
Cordone et al. (1998)	1241	12,5	995	2,91	1408	12,3	1139	3,38	834	10	591	3
Caseau et al. (1999)	1233	12,42	990	3,09	1403	12	1220	3,38	828	10	596	3
Bräysis (2001a)	1253	12,17	1039	2,82	1408	11,88	1244	3,25	832	10	593	3
Zhu y Lee (1999) 2-int	1469		1330		1680		1700		965		780	
Guillén (2002)	1955	26	**1239**	8	2247	21	**1573**	11	1955	25	1485	14

Tabla 4.17 Comparación con los métodos de mejora

También merece la pena destacar en este punto la preferencia de algunos autores por algún tipo de problemas, como es el caso de Shaw (1997 y 1998). En este sentido se observa ya en este punto que algunos autores optan por aportar soluciones solamente a algún grupo de problemas, poniendo en tela de juicio la robustez de sus métodos para

resolver cualquier tipo de problemas. Lo mismo ocurre cuando prestamos atención a algunos de los métodos más avanzados del tipo Tabu Search o de Algoritmos Genéticos, tal y como se muestra en las tablas siguientes.

Los algoritmos de tipo Tabu Search suelen partir de una solución inicial creada a través de alguno de los métodos de construcción expuestos en las líneas anteriores, y después se aplica alguno de los procedimientos de mejora comentados anteriormente, pero con la particularidad de que en vez de perseguir una aplicación de las reglas de mejora de forma aleatoria, en el caso de la búsqueda tabú se presentan unas reglas para la aplicación de las mismas, de manera que se penalizan algunos de los movimientos ya realizados para que el algoritmos en cuestión huya de los óptimos locales, para buscar en nuevos espacios. Una vez más la eficacia de estos métodos dependen de la solución inicial obtenida, y de la eficacia de los procedimientos de mejora aplicados.

Al igual que en el caso anterior una comparativa directa de este tipo de métodos con los métodos de construcción de rutas no tiene sentido debido a la baja eficacia de los mismos, y a la potencia de búsqueda de los procedimientos de Búsqueda Tabú. Sin embargo también se ha optado por incluir una representación de los resultados obtenidos por este tipo de métodos con la intención de mostrar su potencial, y las mejoras conseguidas a través de estas búsquedas intensivas, sobre las soluciones inicialmente generadas por los procedimientos de construcción.

Búsqueda Tabú	**R1**		**R2**		**RC1**		**RC2**		**C1**		**C2**	
	Dist	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**
Garcia et al. (1994)	1320	12,9	1229	3,1	1483	12,9	1551	3,9	877	10	602	3
Rochat y Taillard (1995)	1197	12,6	954	3,1	1370	12,4	1140	3,6	828	10	589	3
Potvin et al. (1996a)	1295	12,6	1186	3,1	1465	12,6	1476	3,4	850	10	594	3
Taillard et al. (1997)	1220	12,3	1013	3	1381	12,9	1199	3,4	828	10	589	3
Chiang et al. (1997)	1204	12,17	986	2,73	1397	11,88	1229	3,25	828	10	591	3
De Backer et al. (1997)	1214	14,17	930	5,27	1385	14,25	1099	6,25	829	10	604	3,25
Brandao (1999)	1205	12,58	995	3,18	1371	12,13	1250	3,5	829	10	591	3
Schulze y Fahle (1999)	1268	12,5	1056	3,1	1396	12,3	1308	3,4	828	10	589	3
Tan et al. (2000)	1266	13,83	1080	3,82	1458	13,63	1293	4,25	870	10	634	3,25
Lau et al. (2000)	1211	14	960	3,55	1385	13,63	1232	4,25	832	10	612	3
Cordeau et al. (2001)	1210	12,08	969	2,73	1389	11,5	1134	3,25	828	10	589	3
Zhu y Lee (1999)	1292		1097		1471		1331		874		644	
Guillén (2002)	1955	26	1239	8	2247	21	1573	11	1955	25	1485	14

Tabla 4.18 Comparación con los métodos de Búsqueda Tabú

En este caso también merece la pena destacar que la mayor parte de los autores consiguen resolver los problemas de tipo *C* de forma óptima, debido a su baja complejidad, motivado en parte por el hecho de que los nodos están agrupados, y de esta manera se reducen drásticamente el número de buenas combinaciones entre los nodos, de manera que en la solución óptima generalmente los nodos agrupados forman parte de la misma

ruta, por lo que se trata en general de problemas más sencillos que los de tipo *R* y *RC*. Tal y como se apuntó en las líneas anteriores, el modelo recogido en este trabajo opera mejor con aquellos problemas en los que los nodos tienen una mayor dispersión, de manera que el mecanismo de inserción diseñado adquiere su mayor potencial de ahorros en las inserciones, encontrándose con aquellos nodos intermedios para realizar visitas intermedias en los largos recorridos.

Los últimos avances encontrados en los métodos de cálculo de soluciones, implementan siempre las técnicas de cálculo recogidas en los algoritmos genéticos, donde generalmente se cuenta con un conjunto muy amplio de soluciones. Estas soluciones se obtienen de los métodos de construcción de rutas, a través de dos vías diferentes. Algunos autores optan por obtener un elevado número de soluciones iniciales a través de variaciones en los parámetros de los algoritmos de construcción de rutas, capaces de arrojar diferentes resultados ante variaciones en sus parámetros de cálculo. Sin embargo otros autores optan por generar una única solución inicial, y a partir de ésta, aplicando los procedimientos de mejora, obtener otras tantas variantes para constituir la población inicial para el algoritmo.

Sobre esa población inicial se aplican preferentemente tres tipos de reglas derivadas de los principios genéticos en los que se inspira el algoritmo. La primera regla consiste en la *selección* de aquellas combinaciones de genes mejores, que se corresponden con uniones de arcos integradas en las rutas de esas soluciones y que el algoritmo las considera como buenas uniones. A partir de la selección de esas secuencias de nodos, se aplican los procedimientos de *reproducción* o combinación, para formar nuevas soluciones a partir de las soluciones padre, de manera que se generan soluciones con nuevas rutas, llamadas descendientes, y que dan lugar a las nuevas generaciones de soluciones. Y por último, en el caso de que el algoritmo se estanque en zonas correspondientes a óptimos locales, se aplican las reglas de *mutación*, que implican cambios sobre una solución determinada a través de alteraciones en los genes o secuencias de nodos de esas soluciones.

Generalmente las soluciones obtenidas a través de los algoritmos genéticos se encuentran entre las soluciones mejores y más robustas, principalmente motivado por las estrategias de búsqueda intensiva empleadas, pero que obedecen fundamentalmente a criterios matemáticos, y en las que la lógica intrínseca no es nada aparente, por supuesto mucho menos evidente que en los procedimientos de construcción de rutas expuestos al comienzo, y donde la lógica, tiene que ver mucho con el tipo de decisiones implementadas en los departamentos de planificación de rutas de las empresas de transporte.

A pesar de la elevada eficacia de estos métodos, cabe destacar una vez más la gran dependencia que los mismos tienen sobre los métodos de construcción, y los procedimientos de mejora, ya sean a través de mecanismos sencillos como el 2-opt, 3-opt, Intercambio, etc. o incluso los más avanzados de búsqueda tabú.

Algoritmos Genéticos	R1	R2	RC1	RC2	C1	C2

	Dist	NV	Dist	NV	Dist	NV	Dist	NV	Dist	NV	Dist	NV
Thangiah (1995a)	1300	12,75	1124	3,18	1474	12,5	1411	3,38	892	10	749	3
Potvin et al. (1996)	1296	12,58	1117	3	1446	12,13	1368	3,38	838	10	590	3
Berger et al. (1998)	1261	12,58	1030	3,09	1441	12,13	1284	3,5	834	10	594	3
Bräysy (1999b)	1272	12,58	1053	3,09	1417	12,13	1256	3,38	857	10	624	3
Homberger y Gehring (1999)	1228	11,9	967	2,7	1393	11,6	1144	3,3	828,38	10	589	3
Homberger y Gehring (1999)	1226	12	1034	2,7	1407	11,5	1176	3,3	828,38	10	589	3,25
Gehring et al. (1999)	1198	12,42	947	2,82	1356	11,88	1140	3,25	828,38	10	590	3
Gehring et al. (2001)	1208	12,08	965	2,73	1389	11,5	1126	3,25	828,5	10	590	3
Gehring et al. (2001)	1217	12	961	2,73	1395	11,5	1139	3,25	828,63	10	590,3	3,25
Bräysy et al. (2000)	1213	12,42	978	3,09	1372	12,13	1170	3,38	828	10	591	3
Berger et al. (2000)	1221	11,92	975	2,73	1389	11,5	1159	3,25	828	10	590	3
Tan et al. (2000)	1314	14,42	1093	5,64	1512	14,63	1282	7	860	10,11	589	
Wee Kit et al. (2001)	1203	12,58	951	3,18	1382	12,75	1132	3,75	833	10	593	
Rahoual et al. (2001)	1362	12,92			1487	12,63			887	10		
Guillén (2002)	1955	26	1239	8	2247	21	1573	11	1955	25	1485	14

Tabla 4.19 Comparación con los métodos de Algoritmos Genéticos

A pesar de ser técnicas poco utilizadas para la resolución del problema VRPTW, también se han recogido en este apartado las soluciones arrojadas por algunos de los métodos basados en el Recocido Simulado. En este sentido cabe destacar la baja eficacia generalmente mostrada por este tipo de métodos en contra de aquellos que utilizan la búsqueda Tabú, o los Algoritmos Genéticos. Esta baja eficacia deriva principalmente de los fundamentos del Recocido Simulado, ya que se trata de una técnica de búsqueda fundamentada en procesos altamente estocásticos, de manera que la búsqueda salta de forma aleatoria de unas zonas a otras, provocando cambios artificiales en las soluciones, pero sin obedecer a ningún tipo de criterio lógico. Si bien la búsqueda tabú penalizaba algunos movimientos por resultar repetitivos, el recocido simulado aplica aleatoriamente técnicas de mejora local como las 2-opt, 3-opt, Or-opt, etc., y de esta manera explora otras zonas del espacio de soluciones.

Generalmente esta aleatoriedad implica largos tiempos de computación, muchas veces improductivos, ya que generalmente las buenas soluciones tienden a estar agrupadas, tal y como se apreciaba en el ejemplo del problema R103 expuesto en el apartado anterior, por lo que estos saltos aleatorios suponen búsquedas en ese espacio a modo de "palos de ciego", ya que se prueban múltiples opciones sin ningún tipo de lógica, sin simplemente a través de búsquedas intensivas. Estos procedimientos suelen ser tanto peores cuanto más concentradas están las buenas soluciones, lo que ocurre en los problemas de tipo *C*. En este sentido los peores resultados de este tipo de métodos se obtienen en los problemas aparentemente más fáciles de resolver, tal y como ocurre con el algoritmo expuesto en este trabajo. Estas observaciones se concretan en la tabla 4.20.

Recocido Simulado	**R1**	**R2**	**RC1**	**RC2**	**C1**	**C2**

	Dist	NV	Dist	NV	Dist	NV	Dist	NV	Dist	NV	Dist	NV
Thangiah et al. (1994)	1227	12,33	1005	3	1391	12	1173	3,38	830	10	640	3
Chiang et al. (1996)	1308	12,5	1166	2,91	1473	12,38	1393	3,38	909	10	666	3
Tan et al. (2000)	1420	14,5	1278	3,64	1648	14,75	1641	4,25	958	10,11	766	3
Zhu y Lee (1999) SA	1422		1279		1657		1642		943		766	
Guillén (2002)	1955	26	**1239**	8	2247	21	**1573**	11	1955	25	1485	14

Tabla 4.20 Comparación con los métodos de Recocido Simulado

En este caso el algoritmo recogido en este trabajo también mejora alguno de los métodos fundamentados en la técnica del Recocido Simulado.

Por último se recogen en la tabla 4.21 los resultados de otros algoritmos recogidos en el capítulo 2, y que utilizan diferentes metodologías para la resolución del VRPTW, entre las que destacan las Colonias de Hormigas, Redes Neuronales, y otros..

En este caso también se mejoran muchos de los ejemplos recogidos en esta tabla, sobre todo en el caso de los problemas *R2* y *RC2*, donde la aplicación del método desarrollado en este trabajo supera a algunos de los métodos de cálculo más potentes recogidos en la literatura.

OTROS	**R1**		**R2**		**RC1**		**RC2**		**C1**		**C2**	
	Dist	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**	**Dist**	**NV**
Kontoravdis y Bard (1995)	1325	12,6	1164	3,1	1501	12,6	1414	3,5	827	10	590	3
Potvin et al. (1995)	1539	13,58	1325	3,09	1828	13,63	1578	3,63	1237	10,56	875	3,38
Bachem et al. (1996)	1417	12,8	1250	3,1	1532	12,5	1512	3,4				
Liu et al. (1999)	1249	12,17	1016	2,82	1412	11,88	1204	3,25	830	10	591	3
Kilby et al. (1999)	1200	12,7	966,6	3	1388	12,1	1333	3,4	830	10	591	3
Caseau et al. (1999b)	1207	12,17			1356	12						
Gambardella et al. (1999)	1211	12,4	960	3	1388	11,9	1149	3,3				
Riise et al. (1999)	1211	13,92	917	4,91	1399	13,75	1055	5,63	846	10,56	598	3,88
Rousseau et al. (2000)	1210	12,08	941	3	1382	11,63	1105	3,38	828	10	589	3
Anderson et al. (2000)					1397	11,63						
Guillén (2002)	1955	26	**1239**	8	2247	21	**1573**	11	1955	25	1485	14

Tabla 4.21 Comparación con otros métodos

En resumen, en este apartado se muestran a modo de comparativa los resultados medios obtenidos por los diferentes métodos desarrollados hasta el momento y que recogen la mayor parte de los trabajos en el ámbito de la problemática del VRPTW. En este sentido, la comparativa de estos métodos ha de ser intragrupo, dadas las diferencias en los métodos de cálculo expuestos anteriormente. También es necesario resaltar la ineficacia de alguno de los métodos tradicionalmente aplicados en otras problemáticas similares, como ocurre con el Recocido Simulado, ampliamente utilizado en el caso del VRP.

La supremacía de los Algoritmos Genéticos rivaliza actualmente con la incipiente línea de investigación fundamentada en las Colonias Artificiales de Hormigas, aunque ambas contienen principios muy similares.

Por último, destacar la dependencia que la mayor parte de los métodos tienen sobre los métodos clásicos de construcción de rutas, dada la complejidad de los mismos. En este sentido los autores prefieren desarrollar diferentes métodos de implementación de procesos de búsqueda intensiva, ya sean algoritmos genéticos, búsqueda tabú, recocido simulado, o cualesquiera otros, que no desarrollar nuevos métodos de construcción a partir del conjunto de nodos existentes. De esta manera se ha generado una elevada dependencia de los métodos clásicos, principalmente el método *I1* de Solomon, que se halla presente en la mayor parte de los trabajos.

La eficacia de las soluciones obtenidas por los métodos metaheurísticos dependen de la eficacia mostrada en los métodos de construcción, y en este caso también de la eficiencia de los mismos, con respecto al tiempo de computación. Por ello las mejoras en estos métodos de construcción pueden aportar grandes avances en los métodos metaheurísticos.

Por esto se ha querido profundizar en los mecanismos de construcción de rutas con la intención de crear un nuevo método cuya eficacia se demuestra tanto en los resultados obtenidos para algunos de los grupos de problemas clásicos (en concreto *R2* y *RC2*), como en la rapidísima velocidad de cálculo de soluciones. Por ello los futuros desarrollos sobre este método, así como su implementación a través de procesos de búsqueda tabú, y quizás con resultados mucho más prometedores, a través de los algoritmos genéticos, puedan reportar en futuras investigaciones y desarrollos, importantes reducciones tanto en las distancias totales recorridas, como en el número de rutas finales de la solución.

3.2 Comparativa de las velocidades de cálculo y análisis de eficiencia

En el presente apartado se comentan con mayor detalle las velocidades de cálculo de algunos algoritmos analizados anteriormente. Igualmente se presenta una comparativa de la velocidad de cálculo del algoritmo desarrollado en este trabajo.

Antes de abordar esta tarea, volvemos a recordar que la finalidad del algoritmo no persigue superar los mejores resultados conseguidos hasta el momento, por lo que las consideraciones con respecto a los mediocres resultados obtenidos para la distancia total acumulada, solamente deben tenerse en cuenta a efectos demostrativos.

La intención del presente trabajo ha sido hasta el momento analizar los procesos de construcción de rutas de transporte en el problema del VRPTW, para lo que se han analizado multitud de métodos y técnicas constructivas para conseguir buenas rutas, La mayor parte de estos métodos se basan en búsquedas intensivas exhaustivas, mientras que el método aquí planteado busca unas reglas deterministas para la construcción de rutas de transporte. La gran variedad de problemas en la literatura, obliga a múltiples criterios de decisión, muchas veces contrapuestos, de manera que lo que aparentemente es

beneficioso para un tipo de problemas, no lo es tanto para otros. Por esto a nivel global, el comportamiento de los métodos de búsqueda intensiva siempre rinde mejores resultados que el método aquí recogido. Estas afirmaciones se desprenden de la figura 4.8, elaborada a partir de los datos recogidos en Bräysis y Gendreau (2001)[21].

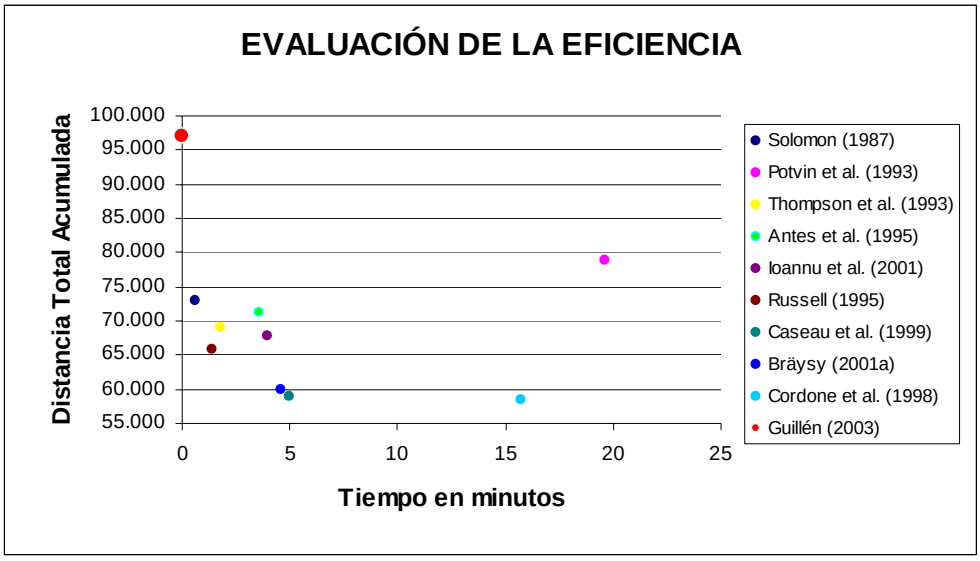


Figura 4.8 Representación de los valores de las distancias totales acumuladas, y tiempos de cálculo medios

A pesar de ello, generalmente estos resultados provocan períodos de computación más elevados que los del algoritmo aquí desarrollado, por lo que merma su eficiencia a consta de una mayor eficacia. No se han incluido en la figura anterior los resultados obtenidos por los métodos de búsqueda tabú, ni recocido simulado, ni los algoritmos genéticos. Esta decisión obedece, en primer lugar, a que los tiempos medios para cualquiera de estos métodos, suelen ascender a 4 ó 5 horas de computación, y en ocasiones requieren de computación paralela en múltiples ordenadores. Por ello su inclusión en la figura anterior hubiera distorsionado las diferencias entre los métodos de construcción y mejora representados.

De entre los algoritmos metaheurísticos analizados, el que más tiempo invierte en computación es el de Gehring et al. (2001), con un tiempo medio de 1.458 minutos, en una red con 4 procesadores Pentium a 400 Mhz, y unos resultados de 57.641 unidades temporales para la distancia total acumulada. Este ejemplo individual, representa la tónica general de los procedimientos metaheurísticos fundamentados en el cálculo intensivo para la resolución de problemas, y que nada tienen que ver con el algoritmo aquí presentado.

[21] En el trabajo original se presenta la comparativa de todos los métodos tanto a través de la distancia total acumulada como el número de vehículos acumulados. Igualmente se indica las máquinas utilizadas por cada uno de los autores, y las velocidades de cálculo correspondientes.

Cabe destacar una de las conclusiones más esperanzadoras del presente trabajo, y que hace referencia a las amplias posibilidades de mejora del presente algoritmo para obtener mejores resultados, sin perjudicar el tiempo de computación. En el siguiente apartado se detallan las ventajas e inconvenientes del método, así como posibles soluciones a estos inconvenientes, con la intención de fomentar posteriores investigaciones sobre el método, para lo cual también se aportan las secuencias de código originales.

El proceso de análisis de las decisiones de inserción simple, inserción compuesta, adición, y la limitación de las zonas de influencia, ha permitido obtener información de alto valor para seguir trabajando en mejoras y futuros desarrollos del método, que sin duda permitirán obtener importantes avances en la aplicación de reglas deterministas para la construcción de rutas, en contra de los procesos de búsqueda aleatoria, o de cálculo intensivo. De esta manera se ganará en eficiencia, eficacia, y sobre todo en el determinismo de los propios métodos de construcción, en contra de la aleatoriedad recogida en algunos planteamientos.

4 Conclusiones

En este apartado se destacan las conclusiones relativas al desarrollo y funcionamiento del algoritmo presentado en este trabajo. Para ello se exponen tanto los puntos a favor, como los inconvenientes relativos a la eficacia y eficiencia del mismo, así como un resumen de las aportaciones generales del mismo al estado de la cuestión.

4.1 Ventajas del algoritmo

Tras haber implementado el algoritmo heurístico desarrollado en este trabajo sobre la batería de problemas tipo presentados por Solomon (1987), podemos afirmar desde un punto de vista global, que le modelo aquí recogido supone una nueva alternativa en el marco de los algoritmos de construcción de rutas, aportando un nuevo modelo y una nueva filosofía en el problema de la planificación de rutas de transporte con ventanas de tiempo.

Las principales ventajas del algoritmo radican en su eficiencia global, su rapidez de cálculo, su flexibilidad, su determinismo, y por último, su parametrización.

4.1.1 Eficiencia global

En el caso de los algoritmos de resolución del problema del VRPTW, es necesario tener en cuenta tanto la eficacia en la resolución de problemas, como los recursos consumidos para obtener esas soluciones, principalmente a través del cómputo del tiempo total necesario para obtener esas soluciones. En este sentido el algoritmo presentado en este trabajo es de los algoritmos más rápidos desarrollados hasta la fecha, siendo capaz de generar en un ordenador con un procesador Pentium IV a 1,7 Gh, un total de 4.000.000 soluciones en poco más de dos horas, siendo el tiempo necesario para una sola solución de unos 0,02 segundos. Esta velocidad se alcanza debido a la búsqueda restringida integrada en el algoritmo. El algoritmo también se ha implementado en ordenadores con procesadores mucho más lentos, sin que el tiempo necesario para una iteración individual se incrementase alarmantemente. Por ello se presta a la aplicación del mismo en la generación de poblaciones iniciales por ejemplo, para la implementación de métodos basados en algoritmos genéticos, sin que esto suponga grandes desfases temporales.

En cuanto a los resultados arrojados por el algoritmo cabe destacar la eficacia lograda para alguno de los grupos de Solomon, en concreto los *R2* y *RC2*, donde se combina la existencia de nodos libres con ventanas de tiempo unilaterales, con aquellos que constan de ventanas de tiempo bilaterales, e igualmente los nodos se encuentran dispersos sin que la agrupación de los mismos en pequeños clusters suponga excesivos agravantes para el problema. Principalmente se muestra muy eficaz en la resolución de problemas donde el número de servicios es elevado, y además se encuentra en zonas donde los tiempos de desplazamiento son reducidos. Por ejemplo para los servicios en una ciudad, o en una provincia, de manera que las rutas incluyen muchos servicios, y el número de vehículos es reducido. Sin embargo, cuando el horizonte de planificación es más breve, en

comparación con los tiempos de desplazamiento entre nodos, suele ser más perjudicial para este algoritmo. Por ello podemos resumir, que este algoritmo se presenta como una buena alternativa para aquellos casos donde la planificación supone muchas visitas para cada vehículo, en espacios geográficos reducidos, y donde coexisten clientes con ventanas de tiempo bilaterales y unilaterales. Igualmente el algoritmo ha mostrado su preferencia por las ventanas de tiempo homogéneas, e igualmente los tiempos de servicio similares, aunque el incumplimiento de estas dos circunstancias no impida la resolución del problema. Además estas dos hipótesis son fácilmente asumibles en el caso de la planificación en circunstancias reales, ya que generalmente las empresas utilizan un tiempo medio estimado para cada servicio, y además las ventanas de tiempo suelen implicar amplitudes muy similares. Un dato que sí requiere el algoritmo de los problemas que resuelve mejor, es que las ventanas de tiempo han de ser preferiblemente superiores a los tiempos de servicio, de manera que se aporta una mayor flexibilidad al problema. Por último reseñar, que una excesiva flexibilidad es contraproducente, sobre todo en cuanto a la ausencia de ventanas de tiempo, que como es sabido sirven de guía en la elección tanto de los nodos semilla, como de los sucesivos nodos a ser integrados en las rutas.

4.1.2 Flexibilidad

Una de las características más valoradas por los autores en la literatura, es la flexibilidad de los algoritmos desarrollados para resolver cualquier tipo de problemas. No solamente nos referimos a la robustez de los resultados, que implica cierta homogeneidad en la consecución de buenos resultados, sino a la posibilidad de resolver problemas con cualquier tipo de características en cuanto al número de nodos, número de vehículos, capacidades etc. En este sentido el algoritmo aquí presentado se ha limitado a los condicionantes estipulados en los problemas de Solomon, con la única intención de obtener un marco de referencia. A pesar de ello el método es fácilmente generalizable para contemplar un número ilimitado de nodos, un número ilimitado de vehículos, así como la modificación de cualquiera de las características relativas a los nodos en consideración. Por ello, la adaptación del algoritmo a cualquier necesidad, es una cuestión sencilla de alcanzar, y que requieren de leves modificaciones en el código, algunas de las cuales ya han sido contempladas para validar su aplicabilidad a situaciones reales (i.e. posición de los nodos, modificación de las ventanas de tiempo, demandas de los clientes, número de nodos, y número de vehículos disponibles, así como sus capacidades, y el horizonte de planificación, marcados por los momentos de apertura y cierre del depósito).

Sin embargo todavía es necesario profundizar en nuevas reglas de decisión específicas para los casos en los que los nodos se encuentran agrupados. Estos casos se corresponden con los problemas tipo *C*, que eran en los que el algoritmo arrojaba los peores resultados, parte por su baja robustez, y parte por la optimalidad fácilmente alcanzada por los demás algoritmos, lo que provocaba excesos de la distancia sobre los mejores resultados, muy elevados.

4.1.3 Determinismo

Tal y como se apuntaba al comienzo del desarrollo del algoritmo, se trata de un método determinista, lo cuál implica que goza de todas las ventajas y desventajas de los mismos.

A pesar de todas ellas, la característica que más apuntala la utilización de este tipo de métodos es la posibilidad de replicar cualquier tipo de solución, conociendo de antemano los parámetros, algo que no ocurre con los algoritmos de tipo estocástico o aleatorio, como la búsqueda tabú, el recocido simulado, o los algoritmos genéticos. En este sentido también reporta una garantía de robustez que no aportan los algoritmos estocásticos o aleatorios.

4.1.4 Parametrización

El hecho de tratarse de un algoritmo parametrizado también permite establecer las mejores circunstancias de búsqueda para cualquier tipo de problema, ya que como se ha visto en los apartados anteriores, cada problema implica unas características intrínsecas completamente diferentes, de manera que podríamos afirmar que cada uno se ha de resolver utilizando reglas diferentes de decisión, tan propias de cada problema que las generalizaciones de las mismas son difíciles de obtener, algo que se ha intentado perseguir con este método y que en general los algoritmos recogidos en la literatura, lo resuelven con búsquedas intensivas a modo de "*prueba y error*". En este trabajo se ha procurado buscar una cierta lógica en las decisiones que deben de tomarse por parte de un algoritmo de planificación de rutas, y que traten en la medida de lo posible de asemejarse a las decisiones que tomaría un planificador, de forma que la construcción de rutas, aunque automatizada, sigue criterios lógicos de decisión, y no búsquedas intensivas de combinaciones a través de múltiples y numerosas pruebas. La parametrización en este caso persigue una doble finalidad, por un lado permite la adaptación del método a cada problema particular, tal y como se ha visto anteriormente, y en segundo lugar, genera un elevado número de soluciones de un único problema, que se pueden utilizar en aplicaciones de algoritmos genéticos o cualquier otro tipo de búsqueda metaheurística.

En resumen, podemos afirmar que se trata de un algoritmo nuevo para la construcción de rutas, que persigue la asimilación de las reglas de decisión empresariales para la construcción de rutas de transporte. En ese sentido se ha mostrado su nivel de eficacia y eficiencia en los problemas tipo de la literatura, y se han detectado algunos inconvenientes en la forma de búsqueda codificada en el algoritmo, tal y como se recoge a continuación. Por ello en el siguiente apartado se destacan los puntos débiles del algoritmo, así como posibles formas de corrección que darían lugar a nuevos y más potentes algoritmos de construcción, con búsquedas controladas a través de diferentes filtros. En este sentido se ha aprendido a través de los fallos contenidos en el presente algoritmo, nuevas formas de afrontar el problema del VRPTW.

4.2 Inconvenientes del algoritmo

A pesar de las ventajas reseñadas en el apartado anterior, y de los resultados conseguidos en algunos de los ejemplos de Solomon (1987), el algoritmo también consta de múltiples inconvenientes, algunos de ellos fácilmente salvables. En este apartado se comentan y describen todos los puntos débiles detectados en el algoritmo en comparación con los algoritmos estudiados en el presente trabajo. Para estos inconvenientes se indican también posibles mejoras o mecanismos para solventarlos.

Entre los puntos débiles del algoritmo se incluyen los siguientes:

4.2.1 Selección y asignación de los primeros nodos semilla.

En el procedimiento de selección de nodos iniciales para constituir las primeras rutas, nos encontramos con que el algoritmo contempla la asignación de cada uno de los nodos a rutas independientes, con la intención de que cada uno de estos nodos genere una nueva ruta. Sin embargo en ocasiones estos nodos también son susceptibles de formar parte de la misma ruta, debido a su secuencialidad en ventanas de tiempo, o bien debido a su cercanía. De esta manera se están limitando uniones que en ocasiones pueden resultar beneficiosas para la solución final.

Solución

Una posible solución para resolver este problema, es la posibilidad de que para cada uno de los nodos seleccionados como nodos semilla, se estudie la posibilidad de formar parte de las rutas ya creadas, y si esto ocurre, se seleccionaría un nuevo nodo semilla para tener el cómputo global de los nodos inicialmente contemplados como nodos semilla. En este sentido se procedería de forma igual que como se procede en la segunda e ulteriores etapas de planificación.

4.2.2 Incapacidad de identificación de nodos aislados

Durante las diferentes etapas de selección de nodos para sus adiciones a las rutas que se están planificando, nos encontramos con que la selección de los siguientes nodos a ser asignados se realiza según sus momentos de cierre, y se añaden a las rutas a las que estén más cercanos. De esta manera en ocasiones ocurre que determinados nodos (generalmente periféricos) quedan al margen por tener momentos de cierre más tardíos que los nodos que se han seleccionado para ser añadidos. De esta manera los vehículos huyen de estas zonas hacia nodos que requieran su atención, y que bien podrían haber sido asignados a otras rutas, aunque ello hubiera implicado recorrer una mayor distancia.

Solución

Sería conveniente establecer para cada ruta un factor de retención de esa ruta, evitando que los vehículos abandonasen zonas donde todavía existen nodos no servidos. De esta manera se provocaría que los vehículos se mantuviesen en la zona, bien en espera, o bien cubriendo esos nodos, antes que marcharse en busca de nodos con momentos de cierre más tempranos, que serían asignados a otras rutas que no estuviesen tan cerca. Por ello, sería conveniente estudiar más de una posible ruta candidata a la que sería añadido cada uno de los nodos críticos, lo cual se realizaría en última instancia siempre que no se dejasen nodos descolgados.

4.2.3 Incapacidad de anticipación de contingencias

A veces ocurre que durante la selección de nodos, el algoritmo detecta múltiples nodos con iguales ventanas de tiempo, de manera que es imposible asignarlos a la misma ruta, aunque esté muy cerca el uno del otro. En este sentido se provoca la generación de una nueva ruta, para atender única y exclusivamente al nodo en cuestión.

Solución

En este sentido sería posible implementar en el algoritmo un procedimiento de anticipación de este tipo de circunstancias que provocasen una atracción artificial de dos rutas simultáneas en las etapas anteriores a los momentos de cierre, para que de esta manera cuando llegase esos momentos de cierre, existiesen dos vehículos en las cercanías.

4.2.4 Generación indiscriminada de nuevas rutas.

Dentro del algoritmo, cada vez que es imposible asignar un nodo a una ruta existente, entonces se crea una nueva ruta, que parte del depósito central hacia el nodo en cuestión. De esta manera, sobre todo al final de la planificación, cuando la mayor parte de las rutas están saturadas, o distantes de estos nodos, es cuando se crean la mayor parte de estas nuevas rutas.

Solución

Sería posible un análisis más detallado del proceso de generación de nuevas rutas a través del cuál se estudiase la posibilidad de realizar la adición del nodo con eliminación simple o doble de otros nodos previamente integrados en la ruta. En este sentido se estudiaría la posibilidad de asignar el nodo crítico en la ruta a la que está más cercano, eliminando alguno de los nodos no críticos previamente insertados en esa ruta, de manera que se reduzca o bien la carga de la misma, o bien el contador temporal del vehículo, de forma que se podría servir, o llegar a tiempo al nodo crítico en cuestión. De esta manera se restringe enormemente los procesos de creación de nuevas rutas, provocando finalmente una reducción de las rutas finales de la solución, y probablemente se reducirían también las distancias totales recorridas.

4.2.5 Mantenimiento de los parámetros como valores fijos durante toda la planificación

Una vez seleccionados los valores iniciales de los parámetros R, β y γ, permanecen constantes para todos lo movimientos de adición e inserción de nodos. Esto provoca que se apliquen los mismos tamaños de zonas de influencia para todas las adiciones de nodos a rutas. En este sentido se asume que el tamaño de la zona de inserción, ha de ser igual para todos los posibles movimientos, cuando generalmente no tiene por qué ser así. En este sentido es fácilmente comprensible que en ocasiones será más interesante ampliar las zonas de inserción, y en otras será más interesante disminuir esas zonas de inserción, por lo que deberíamos de contemplar parámetros variables. Por ejemplo, en el caso de

existencia de nodos periféricos, sería conveniente ampliar estas zonas de inserción cuando aparezcan estos nodos periféricos, de manera que probablemente se contemplase su inserción en los siguientes movimientos de los vehículos.

Solución

Una posible solución es hacer variar estos parámetros, fundamentalmente β y γ, de alguna manera, ya sea de forma aleatoria, o bien con respecto a algún criterio lógico, por ejemplo de forma incremental, suponiendo que se comienza con valores muy pequeños, y van aumentando a medida que avanza la planificación, que es cuando menos nodos libres queda para ser insertados. Esta variabilidad permite una posible derivación de este método en procedimientos más potentes de búsqueda tabú o incluso algoritmos genéticos.

4.2.6 Imposibilidad de aprovechar buenas rutas anteriormente contempladas

En el caso de la búsqueda de los valores óptimos de los parámetros, el algoritmo genera una gran cantidad de soluciones integradas por multitud de rutas. Generalmente estas diferentes soluciones integran rutas individuales que pueden identificarse como buenas rutas, dado el elevado número de nodos que las integran, las escasas esperas que se producen, o la saturación de la capacidad de los vehículos. En este sentido, las rutas aparentemente buenas, suelen aparecer en soluciones diferentes. En este sentido se pierden las buenas rutas de algunas soluciones, por soluciones donde la única mejoría es en la distancia total, a veces conseguida con un elevado número de rutas mediocres.

Solución

Se podrían intentar implementaciones del algoritmo que aprovechasen las buenas rutas individuales encontradas durante los procesos de búsqueda para integrarse en una única solución final. En este sentido estaríamos hablando de una aplicación bien de los algoritmos genéticos, o bien de Colonias de Hormigas, para la optimización de las soluciones, aunque en este sentido el objetivo trasciende del propósito inicial de este trabajo.

Tras esta crítica constructiva sobre las ventajas e inconvenientes que plantea el método desarrollado, se procede a presentar las conclusiones alcanzadas durante el curso del presente trabajo y que se recogen en el capítulo 5. Igualmente se plantean posibles líneas alternativas de investigación y desarrollo para los métodos de construcción de rutas de transporte, y en particular para el método propio desarrollado por el autor.

Capítulo 5

Conclusiones y futuras líneas de investigación

En este apartado se recogen las conclusiones sobre la investigación realizada en el presente trabajo, y que recordamos hace referencia a tres áreas complementarias entre sí. En primer lugar se resume las características más destacables sobre los sistemas de planificación de rutas en empresas de transporte, fundamentadas en reglas muy intuitivas aplicadas directamente sobre el problema en cuestión por parte del planificador. En segundo lugar se discuten los problemas más importantes en el desarrollo de soluciones informáticas para el caso español, y que en cierta medida condiciona la aplicabilidad de métodos automáticos de generación de rutas en empresas de transporte. Por último se aportan las conclusiones más relevantes sobre el método desarrollado durante el curso de la investigación realizada, centrando la atención en tres aspectos fundamentales: los datos de entrada para el modelo, las reglas de decisión implementadas, y los resultados arrojados por el mismo.

Por último se detectan las oportunidades de aplicabilidad del método en entornos empresariales, así como aquellas áreas donde es necesario profundizar a través de la persecución de futuras líneas de investigación conducentes al desarrollo de potentes Sistemas de Información Logística.

1 Conclusiones sobre los sistemas de planificación de rutas en las empresas

En este apartado se comentan las conclusiones más destacables sobre la investigación realizada en los sistemas de planificación de rutas en empresas con problemas de distribución. Para ello se ha dividido este apartado en dos grandes bloques; en primer lugar se comenta la situación de los sistemas de decisión en las empresas, sus hipótesis y fundamentos, para destacar a continuación algunos aspectos importantes sobre las soluciones actuales para resolver esta problemática, y que justifican la investigación aquí realizada.

1.1 Sistemas de decisión empresariales

Entre las empresas con problemas logísticos podemos identificar dos grandes grupos:

- Empresas industriales que realizan la distribución física de sus productos a través de una flota que gestiona la empresa.
- Empresas que ofrecen servicios de transporte generales a cualquier empresa o particular.

En el caso de las primeras generalmente se trata de problemas logísticos sencillos de resolver en tanto suelen asociarse a la distribución de las mercancías entre centros propios, o clientes más o menos estables, de manera que las ubicaciones de estos destinos no suelen variar. Generalmente este tipo de planificación de rutas se suele realizar una vez cada mucho tiempo, llegando incluso a contar con rutas fijas en base a zonas o en base a arcos (carreteras). Por ello la planificación necesaria para este tipo de empresas se centra única y exclusivamente en una planificación casi de tipo estratégico o a largo plazo.

Sin embargo en el segundo tipo de empresas nos encontramos con que la incertidumbre de los servicios es mucho mayor, de manera que la necesidad de planificar los envíos o las recogidas, es también más elevada, requiriendo una planificación más operativa de los servicios a través de su programación temporal dentro de la jornada laboral de los conductores que han de realizar esas rutas.

Igualmente cabe destacar para esas empresas, el hecho de que generalmente esta problemática se da en el reparto o recogida de mercancías en zonas geográficas restringidas en las que un vehículo pueda realizar entre 15 y 30 servicios al día, recorriendo pequeñas distancias. Esta circunstancia obedece a que generalmente el transporte denominado de "*larga distancia*" también suele estar prefijado de antemano, atendiendo a una serie de rutas preestablecidas fundamentadas en una red de carreteras existente[22]. Por esto la incertidumbre solamente se da en el reparto final desde los centros

[22] Véase el ejemplo de *VIA CONEXIÓN* en el capítulo 1

de distribución hacia los clientes finales, de manera que el número de servicios, y su localización sí suele variar enormemente de un día para otro.

Por ello se seleccionó esta problemática empresarial para centrar la investigación propuesta en este trabajo, de manera que se buscó un modelo de construcción de rutas para entornos con una gran incertidumbre en los servicios a realizar, y un horizonte de planificación amplio en el que se pudieran realizar por cada vehículo entre 15 y 30 servicios, a unos clientes que están aleatoriamente dispersos en el espacio geográfico indicado. Esta problemática empresarial presenta grandes similitudes con los conjuntos de problemas de Solomon (1987) denominados *R2* y *RC2*, siendo precisamente en estos dos grupos de problemas donde el algoritmo desarrollado muestra un mejor comportamiento, superando a algunos métodos recogidos en la literatura.

La tipología de empresas que se enfrentan a la problemática arriba indicada, generalmente se corresponden con PYMES que actúan de forma independiente en el mercado, o que son delegadas o franquiciadas de otras empresas más grandes, de manera que son las encargadas de realizar la entrega final de las mercancías a los clientes.

Tal y como se recoge en este trabajo se analizaron algunas empresas de estas características, con la intención de aprender de sus métodos y de las reglas de decisión que motivaban la asignación de los envíos a una u otra ruta. En este sentido, es necesario destacar la gran intuición con la que se realizaban estas asignaciones y se construían las rutas por parte del planificador. Estas rutas consistían en secuencias de visitas recogidas en albaranes que debía de cubrir el conductor cuando se entregaba finalmente la mercancía, así como una hoja resumen de las visitas que debía realizar cada conductor. Sin embargo en muchas ocasiones solamente se realizaba la asignación de servicios a un conductor, pero no se indicaba la programación temporal de los mismos, de manera que era el conductor en última instancia quién decidía el orden de los servicios en la zona a la que se había asignado.

Esta planificación, en ocasiones muy exitosa, debido a la experiencia de los participantes, puede resultar ineficiente a medida que se incrementan los servicios a realizar, de manera que aumenta considerablemente la complejidad del problema y el número de combinaciones posibles. Si a esto la añadimos la creciente demanda de servicios sujetos a ventanas temporales, y el aumento de los costes del combustible, entonces nos encontramos con una necesidad acuciante de métodos que faciliten la planificación de rutas de transporte, aportando soluciones de forma eficaz y eficiente, y manteniendo en todo momento los requisitos impuestos por el planificador.

Este problema es conocido por la literatura, y son innumerables los trabajos encontrados en esta línea de investigación que estudian la planificación de rutas de transporte, desde el sencillo problema del viajante o TSP, hasta los más complejos problemas de planificación de rutas de transporte sujetos a restricciones temporales, conocidos como VRPTW. Aunque ampliamente tratado en la literatura, el problema del VRPTW sigue siendo fuente de inspiración para nuevos métodos que persiguen superar récords de velocidad de

cálculo y de calidad de resultados comentados en los diferentes foros sobre optimización. Sin embargo no se ha encontrado hasta la fecha un método que resuelva en tiempo polinómico el problema de forma óptima, ya que se trata de un problema *NP Complete.* Por esto la mayor parte de los trabajos buscan métodos de aproximación a través de algoritmos heurísticos o metaheurísticos.

Es también destacable que la mayor parte de los trabajos realizados utilizan métodos de búsqueda intensiva, como los conocidos Búsqueda Tabú, Recocido Simulado, o los Algoritmos Genéticos; donde generalmente los procedimientos de búsqueda trabajan siempre con un esquema de "*prueba y error*", a través de la consideración de múltiples alternativas generadas con procedimientos de intercambio de nodos entre rutas, y dentro de las mismas.

Sin embargo son muy escasos los trabajos que abordan directamente la construcción de rutas con criterios más lógicos o intuitivos, y que tienen su máximo exponente en Solomon (1987). De hecho una amplia mayoría de autores parten de las soluciones arrojadas por los métodos de Solomon para comenzar sus procedimientos de mejora o de búsqueda intensiva.

La persecución mayoritaria de la línea de investigación basada en metaheurísticos puede ser justificada por la sencillez de aplicación que tienen estos métodos, dado que solamente implican el desarrollo de estrategias de implementación de métodos de optimización matemáticos, fundamentados en modelos de construcción ya existentes. Además es el conjunto de estos métodos los que aportan mejores soluciones, y por ello resultan los más gratificantes para el investigador.

Sin embargo en este trabajo se ha optado por el análisis y desarrollo de nuevas reglas de decisión para la construcción de rutas de transporte, y que han dado lugar al algoritmo que se presenta en este trabajo. Para ello se ha elaborado un método propio y original en el que se parte de una situación de no asignación de ninguno de los nodos en cuestión, y a través de un procedimiento secuencial se van construyendo en paralelo las rutas de la solución final asignando uno a uno los nodos del problema.

La intención de este método no es superar los resultados obtenidos por los procedimientos metaheurísticos descritos anteriormente, sino aprender y contrastar la eficacia de las reglas de decisión desarrolladas para la construcción de rutas. En este sentido una de las conclusiones más importantes de este trabajo, es destacar la escasez de trabajos sobre métodos de construcción de rutas, ya que se reduce a poco menos de una decena, manteniendo enormes similitudes los unos con los otros. Si además tenemos en cuenta que la bondad de los métodos metaheurísticos depende en gran medida de la eficacia de los métodos de construcción en los que se fundamentan, entonces esta necesidad de profundizar en los métodos de construcción de rutas es todavía mayor.

En este trabajo se asume el reto de diseñar nuevas reglas de decisión para la construcción de estas rutas de transporte, de manera diferente a los métodos existentes, ya que la

construcción se realiza atendiendo a los diferentes intervalos temporales en los que se puede dividir el problema, tratando de saturar las rutas para cada uno de los intervalos, y no de forma global, como realiza el método de Solomon (1987). Estas reglas dieron lugar a un método de construcción de rutas que aporta buenos resultados para el tipo de problemas para los que se diseño, y que recordamos eran los de tipo *R2* y *RC2*, donde tanto los resultados obtenidos como los tiempos de computación superan otros métodos de construcción de rutas e incluso algunos métodos metaheurísticos recogidos en la literatura[23]. Aún así no es un método carente de problemas e inconvenientes cuyas soluciones y futuros desarrollos darán lugar a nuevas líneas de investigación sobre las reglas de construcción de rutas. El mayor inconveniente es la escasa robustez del método, ya que su buen comportamiento en los problemas *R2* y *RC2* no es generalizable para otros problemas, especialmente para aquellos en los que los nodos están agrupados en clusters, aunque paradójicamente se trate de los problemas más sencillos de resolver. Igualmente presenta otros inconvenientes que han sido identificados y para los que se aportan posibles soluciones en el capítulo 4, y que podrán abrir nuevas líneas de investigación conducentes a una construcción de rutas más racional y eficiente que en los métodos actuales, y de la que sin duda se beneficiarán los métodos metaheurísticos, para buscar nuevos récords en los resultados de los problemas tipo.

Otra ventaja de los métodos de construcción de rutas suelen guardar una gran similitud con los métodos más intuitivos que desarrollan los planificadores empresariales, de manera que existe una gran analogía entre unos y otros, lo que puede incidir en una mayor aplicabilidad, así como una mayor interactividad entre el método automático de construcción de rutas, y las decisiones y restricciones que el planificador quiera implementar en cada momento.

Esta interactividad con el modelo de planificación de rutas se hace especialmente interesante cuando el planificador también requiere información de la situación en un determinado momento acerca de cómo se está implementando la planificación con la intención de aplicar determinados cambios ante cualquier contingencia o nuevo servicio que se haya de realizar. Esto obedece a los principios de la planificación dinámica, que obliga al decisor a realizar cambios ante cualquier eventualidad, de manera que es necesario reprogramar las rutas existentes para hacer frente a nuevas situaciones. Por ello cabe destacar dos puntos importantes:

- En primer lugar es necesario que exista una interactividad máxima entre el usuario y el método de construcción, y que permita los cambios en las rutas, forzados en ocasiones por el usuario.
- En segundo lugar, estos cambios no han de suponer graves trastornos a la planificación ya ejecutada.

En este punto también cabe destacar las ventajas del algoritmo desarrollado en este trabajo. Generalmente los métodos de construcción de rutas de transporte tienen un

[23] Ver Capítulo 4 para más detalles

enfoque global por cuanto contemplan desde el primer momento la ubicación de todos los nodos, así como los posibles servicios de los mismos, según estén o no abiertos en cada momento. Esto puede llevar a decisiones en las que se rechacen buenas uniones de nodos que por ejemplo están cercanos, por otras en las que se obtiene una mejor distancia total, aún no teniendo esas buenas uniones de nodos. Es decir que se trata de buscar una buena aproximación al óptimo global. En este planteamiento, un pequeño cambio o eventualidad en la planificación puede provocar graves consecuencias en las distancias finalmente recorridas, debido a los escasos márgenes de maniobra que presentan las rutas finales obtenidas de esta manera.

Sin embargo el método desarrollado en este trabajo realiza una optimización por etapas, ya que el horizonte de planificación se divide de forma que en cada uno de los intervalos se tiende a saturar las rutas, con las mejores opciones de entre todos los nodos posibles en ese intervalo, pero que en ningún caso son todos los nodos del problema. En este sentido, simplemente no se contemplan estos nodos, hasta que no están abiertos. Por ello cualquier eventualidad próxima que pueda acontecer, no va a afectar a la planificación ya realizada, simplemente porque en esa planificación esa eventualidad no se habría contemplado, como así hubiera ocurrido en los métodos de construcción clásicos.

Por esto el método desarrollado en este trabajo se presenta como especialmente interesante para la planificación dinámica de rutas de transporte.

1.2 Conclusiones sobre las soluciones actuales para la resolución del VRPTW

Actualmente sólo una pequeña parte de las empresas que sufren la problemática de la planificación de rutas de transporte recurren a la utilización de sistemas informáticos de generación y control de rutas, dado su elevado coste de inversión. Es por esto que generalmente las PYMES suelen recurrir a sistemas de planificación mucho más intuitivos basados en la experiencia del gestor.

La no recurrencia a los sistemas informáticos de planificación de rutas de transporte obedece a la relativa novedad de estos sistemas. Valga comentar el dato de que la mayor parte del software desarrollado para la planificación de rutas de transporte se ha desarrollado en los últimos 5 años, y mayoritariamente en Estados Unidos. Esta puntualización no es caprichosa, sino que tiene un trasfondo importante que deriva de la evolución tecnológica en otras áreas, tal y como se indica en las siguientes líneas.

El desarrollo de software para la planificación de rutas de transporte está teniendo actualmente un despegue importante debido a los avances en múltiples tecnologías complementarias.

- En primer lugar, las velocidades de cálculo de los procesadores. Es conocido por todos los grandes avances en velocidad de computación de los procesadores, así como el abaratamiento de los mismos, lo que han facilitado la implementación de

algoritmos que en otros tiempos se mostraban ineficientes con respecto a los tiempos de computación necesarios. Hoy día la implementación de esos algoritmos ya no supone un problema debido a la potencia de cálculo alcanzada, y que se analiza en algunos trabajos como Dongarra (1998).

- En segundo lugar, los avances conseguidos en los sistemas de información geográfica, o sistemas GIS. Realmente la mayor aplicabilidad de los sistemas de planificación de rutas de transporte se consigue si existen bases de datos con información sobre la red de carreteras, en la que se incluyen los núcleos de población, calles, números, códigos postales, red de carreteras, distancias, y demás información que sustentan los métodos de planificación de rutas. En los últimos años han proliferado las iniciativas para construir estas bases de datos que permitan la generación de mapas vectoriales de estos sistemas GIS. Sin embargo la forma de abordar estas iniciativas ha sido diferente. Mientras que en Estados Unidos, ha sido el propio Gobierno quien ha abordado esta tarea, en otros países como España, han sido empresas privadas. Esto provoca una doble velocidad en los desarrollos de estas aplicaciones. Así por ejemplo, en Estados Unidos, el Gobierno facilita a precios muy reducidos los mapas vectoriales y las bases de datos en sus propios formatos *Tiger,* en España es necesario pagar elevadas sumas de dinero por disponer de las bases de datos que han sido elaboradas por empresas privadas, en ánimo de recuperar la inversión previamente realizada, y minorando el número de posibles desarrollos derivados de esos sistemas GIS. Por esto en España son pocas las iniciativas que se fundamentan en los sistemas GIS, y que en cierta medida inhiben a las empresas y equipos de investigación a perseguir este camino. A pesar de ello recientemente se han detectado algunas iniciativas en este campo, tal y como se comenta en el capítulo 1, por lo que probablemente en los próximos años asistamos a una evolución en los sistemas de geolocalización y geoposicionamiento que permitan un mayor desarrollo de software de planificación de rutas de transporte, que lo haga asequible para las PYMES.
- En tercer lugar, los avances en los sistemas de navegación, y que va unido a lo expuesto en las líneas anteriores relativo a los sistemas GIS. Actualmente se comercializan un sinfín de tecnologías que ayudan a la conducción, denominados sistemas de navegación. Estos sistemas consisten generalmente en módulos de posicionamiento, conectados a bases de datos cartográficas. De manera que es posible visualizar en un mapa vectorial la posición en todo momento de un vehículo. Estos módulos de posicionamiento utilizan generalmente la tecnología GPS. Los sistemas de navegación permiten la introducción de los datos de un destino en concreto, de manera que muestran al usuario todas las direcciones necesarias para llegar a ese destino, según la información sobre la red de carreteras. Estos sistemas son mayoritariamente sistemas *propietarios*, de manera que software y hardware son cerrados impidiendo su adaptación a nuevas necesidades o nuevos desarrollos. Se trata pues de productos comerciales destinados únicamente a esta función.
- Por otro último, cabe destacar el desarrollo en los sistemas de comunicación inalámbrica tipo GSM o UMTS, y que unidos a módulos de geoposicionamiento,

permiten el seguimiento de los vehículos desde una central, de forma que es posible localizar a los vehículos de una flota en cualquier momento. En este sentido la integración de las tecnologías existentes es máxima, dado que confluyen los sistemas de información geográfica, con los sistemas de navegación, y con los sistemas de comunicaciones, de manera que si a esto le añadimos un motor de optimización de rutas, entonces nos encontramos con una potente herramienta de planificación y de ayuda a la toma de decisiones para la empresa de transportes.

Este es el tipo de software más avanzado para la planificación de rutas de transporte que existe en el mercado, y algunos ejemplos de estos productos se han comentado en este trabajo. Sin embargo los elevados costes de adaptación de estos productos al mercado español, provocan una elevadísima inversión y costes de mantenimiento para las empresas, de manera que actualmente solamente una minoría de poco más de una decena de empresas están utilizando este software en España.

Esta motivación también apoya la persecución de esta línea de investigación ya que se detecta en España un enorme potencial para el desarrollo de estos sistemas de planificación.

2 Conclusiones sobre el algoritmo

En este apartado se centra la atención en las reglas de decisión que integran el método heurístico recogido en este trabajo. Para ello se ha dividido esta sección en tres grandes bloques. En primer lugar se realiza una serie de comentarios sobre los datos que incluye el algoritmo, y la forma de tratarlos en las diferentes etapas de planificación, prestando especial atención a la matriz de distancias. En segundo lugar se presentan una serie de reflexiones sobre las reglas de decisión implementadas, y por último se esbozan una serie de conclusiones relativas a los resultados aportados por el modelo

2.1 Consideraciones sobre los datos utilizados.

Para el desarrollo del presente método de aproximación se han utilizado como datos de partida los propuestos en los diferentes ejemplos desarrollados por Solomon (1987), y que pretenden modelizar de alguna manera situaciones reales en las que se presenten problemas asociados al VRPTW.

2.1.1 Localización de cliente y tiempos de desplazamiento

La mayor parte de los algoritmos recogidos en la literatura contrastan su eficacia a través de la resolución de una serie de problemas tipo, mayoritariamente los desarrollados por Solomon (1987), de manera que existen bases de datos con los mejores resultados publicados hasta la fecha. Esto ha llevado a muchos autores a considerar la superación de estos resultados como un fin en sí mismo, desarrollando algoritmos cada vez más potentes para batir los récords obtenidos hasta la fecha.

Sin embargo, no debemos considerar esta investigación como un fin en sí mismo, sino que es necesario que esta investigación tenga una continuidad que provoque el desarrollo de métodos que puedan ser aplicados a situaciones reales, y que de esta manera sirvan a la innovación tecnológica en los procesos empresariales.

Aunque existen múltiples ejemplos de desarrollos obtenidos en base a las investigaciones realizadas por algunos autores, la gran mayoría no contemplan la aplicabilidad de estos resultados, motivado en parte por la excesiva simplificación utilizada en los casos contra los que se contrasta la eficacia de los métodos, y que integran las hipótesis de muchos métodos.

Es por esto que la excesiva simplicidad de los problemas tipo ha guiado a muchos autores en la construcción de sus métodos, de forma que dificultan enormemente la aplicabilidad de estos métodos a situaciones reales.

En gran parte es la hipótesis relativa a la consideración de distancias euclídeas entre los nodos o clientes lo que pone en entredicho la aplicabilidad de estos métodos a situaciones reales. Esta consideración, como es sabido implica que las distancias entre cada par de nodos son euclídeas y simétricas, lo que no se corresponde con la realidad. En este

sentido, para acelerar la aplicabilidad de los métodos propuestos, se llegan a proponer nuevos métodos de cálculos de distancias a través de la implementación de técnicas cuya finalidad es simplificar los cálculos necesarios para reducir tiempos de computación. Los problemas más importantes de esta consideración se recogen a continuación:

- Problema 1: En ocasiones, y debido a la existencia de carreteras o arcos de un solo sentido, nos encontramos con que las distancias de ida y vuelta entre dos nodos son diferentes, por lo que la hipótesis de simetría de la matriz se rompe.
- Problema 2: Las distancias euclídeas son meras simplificaciones de la realidad, no asumibles en una matriz de distancias reales. La hipótesis de distancia lineal entre dos puntos es fácilmente rechazable, y aún más cuando nos encontramos con una orografía complicada como puede ser la del territorio de Galicia. Por ejemplo, si contemplásemos en un problema la distancia lineal entre dos ciudades como La Coruña y Ferrol, nos encontraríamos según las coordenadas de ambas que esta distancia es de 13 Km, cuando la situación real es que la distancia es de aproximadamente 51 kilómetros. Por ello las distancias euclídeas en este tipo de situaciones son irreales.
- Problema 3. Otro inconveniente que nos encontramos a la hora de asumir esta hipótesis en situaciones reales es que en generalmente no son posibles las uniones directas de determinados puntos porque no existen carreteras o vías de unión entre ellos, por lo que una vez más es necesario utilizar otro tipo de vías alternativas que generalmente implican dar grandes rodeos para llegar al punto de destino. En este sentido una vez más la distancia euclidea entre dos puntos es inviable, dado que no se corresponde con la realidad.

Por ello muchos autores defienden la necesidad de contar con datos reales en el desarrollo de métodos de planificación de rutas, por ejemplo a través de los sistemas GIS. De esta manera es posible en todo momento obtener las distancia reales entre cada par de nodos, sin la necesidad de contar con grandes matrices de distancias. En este punto quizás merezca la pena comentar algunos aspectos sobre la obtención de distancias entre nodos a través de estos sistemas.

Dentro de estos sistemas, la obtención de las distancias entre posibles nodos es una actividad trivial:

- Si el sistema permite la geocodificación, entonces bastaría con introducir en el sistema los datos relativos a las direcciones de cada uno de los clientes y del propio almacén, de forma que automáticamente esas direcciones se asocian a sendas ubicaciones en el mapa vectorial integrante del sistema GIS, sobre el que también se representa la red de carreteras existentes. Esta red de carreteras consiste en un conjunto de pequeños tramos, identificados como *objetos*, representativos de los diferentes tramos de una carretera. De esta manera las distancias entre cada par de nodos se obtienen sumando los diferentes tramos que distan entre un cliente y otro, obteniendo de esta manera la distancia total. Para ello se suelen utilizar algoritmos de recorrido mínimo en redes, como el conocido

algoritmo de Dijkstra Sin embargo también es necesario contar con información acerca de las propias carreteras, en las que se identifiquen qué vías son de un sentido, y cuáles de doble sentido, dónde existen núcleos semafóricos, en qué cruces es posible girar hacia sendos lados, etc. Es decir que no solamente es necesario contar con la información relativa a los tramos de las diferentes carreteras, sino que también es necesaria una información que permita la utilización de estas bases de datos como verdaderos sistemas de navegación. Otro problema importante es que por término medio el 95% del tiempo necesario por un algoritmo para arrojar una solución se corresponde con la obtención de las distancias entre cada par de nodos a través de las sucesivas uniones de arcos en esa red de carreteras[24]. El formato más común de estas bases de datos se corresponde con los mapas de *NavTech.*

- Si el sistema no dispone de geocodificación, entonces es necesario ubicar los clientes sobre el mapa vectorial a través de alguna dirección, como pueden ser sus coordenadas espaciales de latitud y longitud, o bien su localización manual por parte del planificador en un mapa vectorial, en el que previamente a identificado la dirección postal del cliente. Una vez realizada esta tarea, entonces el sistema GIS permite el cálculo de las distancias de la manera descrita en el caso anterior.

Hasta la fecha existen muy pocos proveedores de mapas que permitan su utilización como sistemas de Navegación, de manera que generalmente implican fuertes inversiones para las empresas, y para los desarrolladores de software, de forma que se convierten en productos prohibitivos para las PYMES. Es por esto que de momento, y en el caso de España, donde la escasez y elevados precios de estos sistemas todavía se mantiene, no se ha adquirido el nivel de desarrollo tecnológico que si existe en otros países donde los precios de estos sistemas son mucho más reducidos, y la existencia de proveedores de mapas también es elevada.

Por ello en este trabajo se ha mantenido el enfoque de que el cómputo de distancias habría que realizarlo de otra manera que no requiriese la existencia de los sistemas GIS de la zona en cuestión. Para ello se ha mantenido el enfoque de la necesidad de contemplar las distancias reales entre cada dos puntos, pero no obtenida a través de los mencionados sistemas GIS, sino a través de otros medios. Para ello se plantea en este punto la posibilidad de utilizar los tiempos de desplazamiento entre cada par de nodos posibles en la red.

Esta visión es un tanto utópica si hubiera que computar los tiempos entre cada par de posibles destinos para una empresa determinada, y que puede elevarse a cientos de miles de clientes. Sin embargo, esto no tiene por qué se así, ya que podemos restringir de alguna manera estas posibilidades.

[24] Véase Mikkola (2000)

Durante el curso de este trabajo se plantearon diferentes alternativas para la consideración de un sistema de cálculo de distancias entre los posibles clientes de una empresa, y se llegó a la siguiente conclusión.

Es posible utilizar los códigos postales como dato de la localización de un cliente en una red, sin que ello suponga pérdida de realismo.

Ciertamente todas las direcciones postales incluyen un código de zona, de manera que se puede asociar la ubicación de cada cliente con el centro de gravedad del área marcada por ese código sobre un plano. De esta manera es posible la obtención de los tiempos de desplazamiento entre cada par de códigos postales, de forma que se integrarían en una matriz de tiempos de desplazamientos no necesariamente simétrica. De esta manera el número de posibles combinaciones se reduce drásticamente resultando en una matriz manejable, sobre todo si tenemos en cuenta que el problema de distribución de las PYMES a las que hacemos referencia en este trabajo, generalmente es un problema a nivel regional.

De hecho esta consideración es generalmente la que se tiene en cuenta en los sistemas de tarificación por parte de las grandes corporaciones como *UPS* o *FEDEX* donde el precio de un servicio de transporte depende de la distancia existente entre cada par de posibles códigos postales, de manera que el código postal se convierte en una medida de extraordinaria importancia para la determinación de las rutas de transporte.

Esta medida es mucho más aproximada que la consideración de la propia localidad de origen y destino, dado que en localidades grandes podemos encontrar con distancias grandes según la zona del servicio.

Por ello el algoritmo desarrollado en este trabajo ha contemplado la posibilidad de integrar esta posible solución con la intención de incrementar la aplicabilidad futura del mismo. Si bien en los ejemplos resueltos en este trabajo, y que se corresponden con los casos de Solomon (1987), se ha mantenido el enfoque de distancias euclídeas, el desarrollo de las reglas de decisión, mantienen en todo momento las restricciones necesarias para una futura aplicación en circunstancias donde no existan distancias euclídeas.

En resumen, una utilización de la matriz de tiempos de desplazamiento entre cada par de códigos postales puede resolver de forma aproximada la necesidad de contar con datos reales de distancias entre cada par de puntos. Por otro lado, sería necesario contar con una matriz de distancias reales medidas en kilómetros, de manera que el cómputo final del coste de las rutas se realizase según la distancia recorrida, y no el tiempo de desplazamiento. De esta manera los datos de una y otra matriz se complementaría para aportar soluciones mucho más realistas.

Otra indudable ventaja de considerar tiempos de desplazamiento variables es que el usuario o incluso el propio algoritmo puede establecer una retroalimentación que permita

que el algoritmo aprenda de las programaciones que se establecen después de ser ejecutadas. En este sentido, los tiempos de desplazamiento son prefijados y modificados después de completadas las rutas, de manera que en futuras planificaciones se contemplen las posibles variaciones existentes entre los nodos a ser visitados.

En algunas de las entrevistas realizadas con responsables de planificación de rutas destacaban sobre todo el efecto de la estacionalidad existente en determinados nodos como por ejemplo, núcleos de veraneo. En este sentido además de incrementarse las demandas de productos o servicios en estos nodos, provocando un mayor número de visita, los tiempos de desplazamientos a estas zonas también se incrementan motivados por el creciente número de vehículos en estas zonas en los meses de verano.

El hecho de que los tiempos de desplazamientos sean conocidos y manipulables por el usuario reporta grandes ventajas a la hora de plantear la aplicabilidad de los algoritmos a las situaciones reales, permitiendo además que los algoritmos de optimización de rutas, se conviertan en potentes herramientas de inteligencia artificial que apoyen a la toma de decisiones con mayores garantías, al poder adaptarse con el tiempo a la experiencia e intuición de los propios responsables de planificación.

2.1.2 Consideraciones sobre los parámetros del problema

En cuanto a los parámetros limitadores de las zonas de inserción, β y γ, utilizados por el algoritmo es posible disminuir su rango de variabilidad para los ejemplos de Solomon. De esta manera los barridos realizados por el algoritmo reducirían drásticamente los tiempos necesarios, convirtiendo el algoritmo en una herramienta más eficiente.

Por otro lado también es destacable la posibilidad de implementar una cierta aleatoriedad en la elección de los parámetros β y γ. En los casos reportados, se contemplan las mejores soluciones de entre los 4.000.000 de posibles soluciones calculadas para cada problema, sin embargo en cada una de ellas los valores de β y γ han permanecido constante para todos los nodos del problema. Por ello una opción interesante para implementar el algoritmo consistiría en que, para cada nodo seleccionado por el algoritmo para añadirlo a una ruta, los valores de β y γ cambiasen de forma aleatoria. De esta manera se seleccionaría aquella solución mejor de entre todas las reportadas, para la que los valores de los parámetros habrían variado de forma aleatoria en cada uno de los nodos seleccionados. De esta manera las inserciones o no de nodos en el camino se realizaría de forma aleatoria, aceptando y discriminando con criterios diferentes para cada nodo las posibles escalas intermedias entre el nodo origen y el nodo destino.

Otra opción interesante sería hacer variar la evolución de los valores de β y γ según el número de nodos restantes, o del tiempo medio de las rutas, teniendo en cuenta que cuanto más avanzada esté la planificación quedarán menos nodos libres, por lo que los desplazamientos para integrar estos nodos en las rutas cada vez serán superiores, y si β lo impide, entonces finalmente será necesario crear rutas nuevas. Por ello los valores de β y γ debería ser crecientes con respecto al tiempo medio de las rutas.

Estas y otras posibles modificaciones se podrían implementar en el algoritmo con la finalidad de mejorar los resultados del método[25].

2.2 Conclusiones sobre las reglas de decisión del algoritmo

En este apartado se analizan brevemente las reglas de decisión que implementa el algoritmo en cuestión y que se basan en gran medida en los métodos de asignación adición e inserción de nodos.

En este punto cabe destacar que la mayoría de los trabajos recogidos en la literatura realizan un tipo de optimización global en el sentido en que contemplan todos los nodos desde el primer momento, para seleccionar aquellos que mejor se adaptan a cada una de las rutas. Sin embargo en el algoritmo propuesto en este trabajo, los nodos considerados en todo momento son únicamente aquellos cuya visita es posible dentro de la temporalidad de cada ruta o vehículo en cuestión. Es decir que los nodos que no se encuentran abiertos, es como si no existieran. Esto perjudica en gran medida los resultados del algoritmo en los problemas donde el horizonte de planificación es breve, correspondiéndose con los problemas *R1, RC1*, y *C1*, aunque aporta buenos resultados en los problemas donde el horizonte de planificación es más amplio, y por ello el algoritmo cuenta con el tiempo necesario como para que las reglas propuestas muestren todo su potencial.

La ventaja de este algoritmo reside en la forma de construir las rutas, ya que las secuencias de nodos añadidas en cada etapa, generalmente resultan muy buenas secuencias, tomadas individualmente, aunque el resultado final en ocasiones no es tan bueno. Por ello destacamos las siguientes conclusiones sobre las reglas implementadas:

En primer lugar, estas buenas uniones se pueden utilizar por algoritmos de mejora, preferentemente los Algoritmos Genéticos, donde se seleccionarían estas buenas uniones, generadas de manera muy rápida, para formar parte de nuevas rutas. Por esto, y al igual que el resto de los métodos de construcción de rutas, es conveniente su integración en métodos más potentes.

En segundo lugar, dada la forma de optimización de este algoritmo, por etapas, o períodos temporales, se hace especialmente interesante para su aplicación en situaciones de programación dinámica de rutas, dado que se optimiza por tramos, y no de forma global.

En tercer lugar, existe un enorme potencial de mejora de las reglas de decisión implementadas, derivadas de los inconvenientes que plantean estas reglas generales, y que para algunos movimientos, es preferible aplicar determinados filtros correctores. Por ello, el algoritmo, aún habiendo aportado buenos resultados para algunos grupos de problemas tipo, incluso mejores que con otras técnicas más potentes, es necesario desarrollarlo con la intención de incrementar su eficacia.

[25] Véase capítulo 4 sobre Experimentación y Resultados

Además, y tal y como se recoge en el capítulo 4, cabe destacar que se trata de un algoritmo determinista, parametrizado, y muy rápido en el cómputo de soluciones individuales.

2.3 Conclusiones sobre los resultados

Los resultados recogidos en el capítulo 4, muestran la eficacia del algoritmo para resolver los problemas relativos a los grupos *R2*, y *RC2*, cuyas hipótesis y topologías se corresponden en gran medida con la problemática empresarial que se trató desde el comienzo, y que fundamentó el desarrollo del método recogido en este trabajo. Sin embargo cuando aplicamos el algoritmo a otro tipo de problemas, donde las hipótesis cambian, como son los problemas tipo *1*, y los problemas clusterizados, entonces los resultados del algoritmo empeoran. Los motivos y posibles soluciones a esta ineficacia se detallan en el capítulo 4, por lo que no se hace mención expresa en este apartado.

Por término medio, en el caso de los problemas de los grupos *R2* y *RC2*, los resultados arrojados por el algoritmo mejoraban los resultados aportados por otros algoritmos de construcción de rutas, como el Algoritmo de Inserción *I1* de Solomon (1987), incluso se conseguían mejores resultados que otros algoritmos más potentes, fundamentados en técnicas metaheurísticas, como el recocido simulado o algún método de búsqueda tabú.

En resumen, el método presentado y desarrollado en este trabajo consiste en un nuevo método heurístico en el que la construcción de las rutas se realiza de forma paralela, y la optimización se aplica por etapas, no de forma global, y que consigue buenos resultados para los problemas sujetos a horizontes de planificación amplios, donde las rutas contemplan un elevado número de visitas, y las distancias temporales entre cada par de nodos son proporcionalmente pequeñas con respecto al horizonte de planificación, es decir que se trata de clientes que están en áreas geográficas de pequeño tamaño, de forma que cada ruta de la solución sirve entre *20* y *30* clientes. A pesar de ello, existen notables oportunidades de mejora a través de la implementación de filtros en las decisiones de inserción o asignación de nodos que están sujetos a circunstancias especiales, y que provocarían mejoras globales en la solución. Igualmente se recoge la posibilidad de reducir el número de rutas generadas por el algoritmo a través de un posible mecanismo de inserción con eliminación, que no se ha implementado en este algoritmo, pero que dará lugar a futuros desarrollos del mismo, con la intención de mejorar tanto la eficacia del método como su eficiencia computacional.

3 Aplicabilidad del algoritmo a situaciones reales y futuras líneas de investigación.

El presente algoritmo se ha enmarcado dentro de los métodos de construcción de rutas, por lo que sus resultados están en desventaja con respecto a otros métodos mucho más potentes, fundamentados en herramientas de cálculo intensivo. Por ello hablar de su aplicabilidad directa a situaciones reales ciertamente es un ideal. Sin embargo, si es necesario hablar de su potencial de aplicación a situaciones reales una vez se resuelvan dos aspectos fundamentales en el método de cálculo:

- Por un lado es necesario desarrollar algunos filtros en las reglas de decisiones generales del modelo, que permitan una especial consideración de algunos nodos, tal y como se detalla en el capítulo 4, y que sin duda incrementará la eficacia del método.
- En segundo lugar, sería necesario implementar algún mecanismo de mejora, a través de la consideración de intercambios de nodos, del tipo 2-opt, 3-opt, 4-opt, 2-opt*, etc. De manera que se mejore ligeramente la solución finalmente aportada por el método.

Una vez realizada estos desarrollos, y considerando datos reales de distancias entre códigos postales, entonces la aplicabilidad del método a situaciones donde se requiera una planificación de rutas de transporte, sí será una realidad posible.

Para ello sería conveniente desarrollar una aplicación informática que mantuviese en todo momento una interactividad con el usuario, posibilitando y facilitando los cambios que éste desee sobre las rutas generadas por la aplicación, preferentemente a través de métodos de "*arrastre*", mucho más intuitivos que los métodos directos sobre bases de datos.

En cuanto a las futuras líneas de investigación, sería interesante analizar la aplicabilidad de este algoritmo en combinación con métodos metaheurísticos, y en particular, con los Algoritmos Genéticos, ya que dentro de las rutas generadas por el modelo recogido en este trabajo, existen muy buenas uniones que podrían ser aprovechadas por los Algoritmos Genéticos para generar rutas mejores, sobre todo a través del procedimiento de Recombinación o Reproducción.

Igualmente cuando la tecnología derivada de los sistemas GIS sea más asequible para las PYMES, sería interesante estudiar las posibilidades de integrar este método en aplicaciones más potentes donde existan funciones de seguimiento de flotas de vehículos en tiempo real, programación dinámica de las rutas, geocodificación, geolocalización, y geoposicionamiento, etc, de manera que se integren en verdaderos Sistemas de Información Logística, y que unidos a sistemas de información contable, puedan incluso

establecer tarifas, costes y márgenes, convirtiéndose en potentes Sistemas Expertos para la toma de decisiones.

Sin embargo, actualmente la persecución de estas líneas de investigación sigue siendo un área reservada a las grandes corporaciones, debido a las elevadas inversiones necesarias en I+D.

Referencias bibliográficas

1. **Aarts J., Korst H.M. y Van Laarhaven P.J.M**., (1997), "Simulated Annealing", en *Local Search in Combinatorial Optimization*, E. Aarts y J.K. Lenstra, 91-120, John Wiley & Sons, Chichester
2. **Adams J., Balas E. y Zawack D.** (1988) "The shifting Bottleneck Procedure for Job Shop Scheduling" *Management Science* 34, 391-401
3. **Adenso-Díaz B., González M. y García E.** (1998), "A Hierarchical Approach to Managing Dairy Routing", *INTERFACES* 28, 21-31
4. **Anderson D., Anderson E., Lesh N., Marks J., Mirtich B., Ratajczak D. y Ryall K**., (2000), *"Human-Guided Simple Search"*, working paper, Mitsubishi Electric Research Laboratory, Cambridge, U.S.A.
5. **Antes J. y Derigs U.,** (1995), *"A New Parallel Tour Construction Algorithm for the Vehicle Routing Problem with Time Windows"*, working paper, Lehrstuhl fur Wirtschaftsinformatik und Operations Research, Universitet zu Koeln.
6. **Applegate D., y Cook W**., (1991), "A Computational Study of the Job-Shop Scheduling Problem", *ORSA Journal on Computing* 3, 149-156
7. **Applegate D., Bixby R., Ch'vatal V. y Cook W.**, (1995), *"Finding cuts in the TSP"* (a preliminary report). DIMACS Technical Report, 65-85
8. **Bachem A., Hochsättler W. y Malich M** (1996) "The Simulated Trading Heuristic for Solving Vehicle Routing Problems" *Discrete Applied Mathematics* 65, 47-72
9. **Badeau, P., Gendreau M., Guertin F, Potvin J.Y., y Taillard E**., (1995), *"A Parallel Tabu Search Heustic for the Vehicle Routing Problem with Time Windows"*, working paper, CRT-95-84, Centre de Recherche sur les Transport, Universite de Montreal.
10. **Badeau P., Gendrau M., Guertin F., Potvin J.Y. y Taillard E**., (1997), "A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows", *Transportation Research*-C5, 109-122
11. **Baker, E. K., y Schaffer J.R.,** (1986), "Solution improvement heuristics for the vehicle routing and scheduling problem with time window constraints", *Am. J. Math. Mgmnt. Sci*, Vol. 6, p. 261-300.
12. **Balakrishnan N**., (1993), "Simple Heuristics for the Vehicle Routing Problem with Time Windows*", J. Opl. Res. Soc*., Vol. 44, No. 3, p. 279-287.
13. **Barr R.S., Golden B.L., Kelly J.P., Resende M.G.C. y Stewart W.R**., (1995), "Designing and Reporting on Computational Experiments with Heuristic Methods", *Journal of Heuristics* 1, 9-32
14. **Beasley, J. E**., (1990), "Distribution Test Problems by Electronic Mail", *J. Oper. Res. Soc*., Vol. 41, p. 1069
15. **Benyahia I. y Potvin J.Y.** (1995) "Generalization and Refinement of Route Construction Heuristics using Genetic Algorithms" *Proc. Of 1995 IEEE International Conference on Evolutionary Computation,* 39-43, IEEE Service Center, Piscataway, USA
16. **Berge C**. (1970), "*Graphes et Hypergraphes*", Dunond
17. **Berger J., Salois M. y Begin R**., (1998), "A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows", *Lecture Notes in Artificial Intelligence 1418, AI'98 Advances in Artificial Intelligence*, 114-127, Vancouver
18. **Berger J., Barkaoui M. y Bräysy O.,** (2001), "*A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows*", working paper, Defence Research Establsihment Valcartier, Canada
19. **Bland R.G. y Shallcross D.F.** (1989) "Large Traveling Salesman Problems arising from experiments in X-Ray Christallography: a Preliminary Report on Computaiton" *Operations Research Letters*, 8, 125-128

20. **Blanton J.L. y Wainwright R.L.,** (1993), "Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms", en Proceed*ings of the Fifth International Conference in Genetic Algorithms*, S. Forrester (ed), 452-459, Morgan Kaufmann Publishing, San Francisco
21. **Bodin, L. D., y Golden B.L.,** (1981), "Classification in Vehicle Routing and Scheduling", *Networks*, Vol. 11, p. 97-108.
22. **Bodin, L. D. y Levy L**., (1994), "Visualization in Vehicle and Scheduling Problems", *INFORMS J. Comp.*, Vol. 6., No. 3, p. 261-269.
23. **Bodin, L., Golden B.L., Assad A. A. y Ball M.,** (1983), "Routing and Scheduling of Vehicles and Crews, The State of the Art", *Comp. Oper. Res*., Vol. 10, p. 63-211.
24. **Boese K.D.** (1996) *Models for Iterative Global Optimization*, PhD Thesis, University of California, Los Angeles, USA
25. **Bonomi E., y Lutton J.L.,** (1984), "The N-City Traveling Salesman Problem: Statistical Mechanics and the Metropolis Algorithm", *SIAM Review* 26, 551-568
26. **Brady R.M.** (1985) "Optimization Strategies Gleaned from Biological Evolution" *Nature*, Vol 317, No 31, 804-806
27. **Braklow J.W., Graham W.W., Hassler S.M., Peck K.E. y Powell W.B.** (1992) "Interactive Optimization Improves Service and Performance for Yellow Freight System" *Interfaces* 22, 147-172
28. **Bramel J. y Simchi-Levi D.,** (1993), *"Probabilistic Analysis and Practical Algorithms for the Vehicle Routing Problem with Time Windows"*, working paper, Dept. of Indu. Eng. and Oper. Res., Columbia University.
29. **Bramel J. y Simchi-Levi D.**, (1994), *"On the Effectiveness of Set Covering Formulations for the Vehicle Routing Problem with Time Windows"*, working paper, Dept. Indu. Eng. and Manag. Sci., Northwestern University.
30. **Bramel J. y Simchi-Levi D.**, (1995), "A Location Based Heuristic for General Routing Problems", *Operations. Research*, Vol. 43, No. 4, p. 649-660.
31. **Bramel J., y Simchi-Levi D.**, (1996), "Probabilistic Analysis and Practical Algorithms for the Vehicle Routing Problem with Time Windows", *Operations Research* 44, 501-509
32. **Bramel J., Coffman Jr E.G.., Shor P.W. y Simchi-Levi D.**, (1992), "Probabilistic Analysis of the Capacitated Vehicle Routing Problem with Unsplit Demands", *Operations. Research*., Vol. 40, No. 6, p. 1095-1106.
33. **Bräysy O., Gendrau M**., (2001), "*Metaheuristics for the Vehicle Routing Problem with Time Windows*", Internal Report STF42 A01025, SINTEF Applied Mathematics, Department of Optimisation, Norway
34. **Bräysy O**., (1999a), "*A Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows*", Licenciate thesis, University of Vaasa, Finland
35. **Bräysy O.,** (1999b), "A New Algorithm for the Vehicle Routing Problem with Time Windows based on the Hybrization of a Genetic Algorithm and Route Construction Heuristics", *Procceedings of the University of Vaasa, Research Papers* 227
36. **Bräysy O., Berger J., Barkaoui M**, (2000), "*A New Hybrid Evolutionary Algorithm for the Vehicle Routing Problem with Time Windows*", Presented at the Route 2000 workshop, Skoddsborg, Denmark
37. **Bräysy O.**, (2001a), "*Local Search and Variable Neighborhood Search Algorithms for the Vehicle Routing Problem with Time Windows*", Doctoral Dissertation, University of Vaasa, Finland
38. **Bräysy O**., (2001b), "*Five Local Search Algorithms for the Vehicle Routing Problem with Time Windows*", working paper, SINTEF Applied Mathematics, Department of Optimisation, Norway

39. **Bräysy O.,** (2001c), "*A Reactive Variable Neighborhood Search Algorithm for the Vehicle Routing Problem with Time Windows*", SINTEF Applied Mathematics, Department of Optimization, Norway
40. **Bräysy O., y Gendrau M**, (2001), "*Genetic Algorithms for the Vehicle Routing Problem with Time Windows*", SINTEF Applied Mathematics, Department of Optimization, Norway
41. **Breedam, A. Van**, (1994), *"An Analysis of the Behavior of Heuristics for the Vehicle Routing Problem for a Selection of Problems with Vehicle-Related, Customer-Related, and Time-Related Constraints"*, Ph.D. Thesis, Faculty of Appl. Eco., University of Antwerp.
42. **Breedam, A. Van**, (1995), "Vehicle Routing: Bridging the gap between Theory and Practice", *Bel. J. Oper. Res., Stat. & Comp. Sci*., Vol. 35, p. 64-80.
43. **Breedam, A. Van**, (1995b), "Improvements heuristics for the Vehicle Routing Problem based on Simulated Annealing", *Eur. J. Oper. Res*., Vol. 86, p. 480-490.
44. **Carlton W.B**., (1995), "*A Tabu Search Approach to the General Vehicle Routing Problem*", PhD thesis, University of Texas, Austin, U.S.A.
45. **Caseau Y., y Laburthe F.**, (1999), "Heuristics for Large Constrained Vehicle Routing Problems", *Journal of Heuristics* 5, 281-303
46. **Caseau Y., Laburthe F. y Silverstein G.,** (1999), "A Meta-heuristic Factory for Vehicle Routing Problems" en *Principles and Practice of Constraint Programming CP'99, Lecture Notes in Computer Science*, J. Jaffar (ed), 144-158, Springer Verlag, New York
47. **Cerny V.**, (1985), "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", *Journal of Optimization Theory and Applications* 45 (1), 41-51
48. **Chiang W.C. y Russell R.A**., (1996), Simulated Annealing Metaheuristics for the Vehicle Routing Problem with Time Windows", *Annals of Operations Research* 63, 3-27
49. **Chiang W.C. y Russell R.A.,** 1997, "A Reactive Tabu Search Metaheuristic for the Vehicle Routing Problem with Time Windows", *INFORMS Journal on Computing* 9, 417-430
50. **Christofides N.**, (1973), "The Optimum Traversal of a Graph", *OMEGA*, Vol. 1, No. 6, p. 719-732.
51. **Christofides N.**, (1985), "Vehicle Routing", en: *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Lawler, Lenstra, Rinnooy Kan & Shmoys (Eds.), John Wiley, Chichester, UK. p. 431-448.
52. **Christofides, N. y Beasley J.E.,** (1984), "The Period Routing Problem", *Networks*, Vol. 14, p. 237-256.
53. **Christofides N. y Eilon S**., (1969), "An Algorithm for the Vehicle Dispatching Problem", *Oper. Res. Q*., Vol. 20, No. 3, p. 309-319.
54. **Christofides N. y Eilon S**., (1969), "Expected Distances in Distribution Problems", *Oper. Res. Q*., Vol. 20, No. 4, p. 437-443.
55. **Christofides N. y Eilon S**., (1972), "Algorithms for Large-Scale Traveling Salesman Problems", *Oper. Res. Q*., Vol. 23, No. 4, p. 511-518.
56. **Christofides N. y Eilon S.,** (1975), "Algorithms for Large-Scale Traveling Salesman Problems", *Oper. Res. Q*., Vol. 23, No. 4, p. 511-518.
57. **Christofides N., Mingozzi A. y Toth P.,** (1979), "The Vehicle Routing Problem", in: *Combinatorial Optimization*, Christofides, Mingozzi, Toth & Sandi, (Eds.), John Wiley & Sons, Inc. NY., p. 315-338.
58. **Christofides N., Mingozzi A. y Toth P.,** (1981), "State-Space Relaxation of Bounds to Routing Problems", *Networks*, Vol. 11, p. 145-164.

59. **Clarke, G. y Wright J.W.,** (1964), *"*Scheduling of Vehicles from a Depot to a number of Delivery Points", *Oper. Res*., Vol. 12, p. 568-581.
60. **Cordeau J.F., Laporte G. y Mercier A.,** (2001), "A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows", *Journal of the Operational Research Society* 52, 928-936
61. **Cordeau J.F., Gendrau M., Laporte G., Potvin J.Y. y Semet F.**, (2001), "*A Guide to Vehicle Routing Heuristics*", Publication CRT-2001-23, University of Montreal, Canada
62. **Cordone R., y Wolfer-Calvo R.**, (1998), "*A Heuristic for the Vehicle Routing Problem with Time Windows*", Internal Report, Department of Electronics and Information, Polytechnic of Milan, Milan, Italy.
63. **De Backer B., Furnon V, Proesser P., Kilby P., y Shaw P.,** (1997), "*Local Search in Constraint Programming: Application to the Vehicle Routing Problem*", presented at the CP-97 Workshop on Industrial Constraint-based Scheduling, Schloss Hagenberg, Austria
64. **De Backer B., Furnon V., Kilby P., Proesser P., y Shaw P.,** (2000), "Solving Vehicle Routing Problems Using Constraint Programming and Metaheuristics", *Journal of Heuristics* 6, 501-523
65. **Desrochers, M., Lenstra J.K y Savelsbergh M.W.P.**, (1988), "A Classification Scheme for Vehicle Routing and Scheduling Problems", *Eur. J. Oper. Res*., Vol. 46, p. 322-332.
66. **Desrochers, M., Lenstra J.K., Savelsbergh M.W.P., Soumis F.**, (1988), "Vehicle Routing with Time Windows: Optimization and Approximation",en: *Vehicle Routing: Methods and Studies*, B. L. Golden and A. A. Assad (Eds.), Elsevier Sci. Publ. B. V., p. 65-84.
67. **Desrosiers, J., Dumas Y., Solomon M.M. y F. Soumis F.**, (1995), "Time Constrained Routing and Scheduling", en: *Handbooks in Operations Research and Management Science*, Vol. 8, Ball, M. O., T. L. Magnanti, C. L. Monma & G. L. Nemhauser (Eds.), Elsevier Science, Amsterdam, p. 35-140.
68. **Dongarra J**., (1998), "*Performance of Various Computers Using Standard Linear Equations Software*", Report CS-89-85, Department of Computer Science, University of Tennessee, U.S.A.
69. **Duchessi P., Belardo S., y Seagle J.P.**, (1988), "Artificial Intelligence and the Management Science Practitioner: Knowledge Enhancements to a Decision Support System for Vehicle Routing" *INTEFACES* 18:2, 85-93
70. **Dullaert W.,** (2000a), "Impact of Relative Route Length on the Choice of Time Insertion Criteria for Insertion Heuristics for the Vehicle Routing Problem with Time Windows", en *Proceedings of the Rome Jubilee 2000 Conference Improving Knowledge and Tools for Transportation and Logistics Development*: 8th Meeting of the Euro Working Group Transportation, Faculty of Engineering, "La Sapienza", University of Rome, Italy, B. Maurizio (ed), 153-156, Rome
71. **Dullaert W.**, (2000b), "*A Sequential Insertion Heuristic for the Vehicle Routing Problem with Time Windows with Relatively Few Custimers per Route*", Research Paper, Faculty of Applied Economics UFSIA-RUCA, 2000:014, Antwerpen, Bélgica
72. **Duncan T.**, (1995), *"Experiments in the use of Neighbourhood Search techniques for Vehicle Routing"*, working paper, Artif. Intell. Appl. Inst., University of Edinburgh.
73. **Duncan T.**, (1995), *"Schedule-IT: An Intelligent Vehicle Scheduling System"*, working paper, Artif. Intell. Appl. Inst., University of Edinburgh.
74. **Eilon S.,Watson-Gandy C.D.T. y Christofides N.**, (1971), *"Distribution Management': Mathematical Modelling and Practical Analysis"*, Hafner Publ. Co. NY.
75. **Erkut E.y MacLean D.** (1992) "Alberta's energy efficiency branch conducts transportation audits" *Interfaces* 22:3 pp:15-21

76. **Fernández de Córdoba P., García-Raffi L.M., Mayado A. y Sanchís J.M.** (2000) A Real Delivery Problem Dealt with Monte Carlo Techniques. *TOP*, Vol 8,
77. **Fischer M.L. y Jaikumar R.**, (1981), "A Generalized Assignment Heuristic for Vehicle Routing", *Networks*, Vol. 11, p. 109-124.
78. **Fischer M.L., Greenfield A.J., Jaikumar R. y Lester III J.T**., (1982), "A Computerized Vehicle Routing Application", *INTERFACES*, Vol. 12, No. 4, p. 42-51.
79. **Foisy C., y Potvin J.Y.** (1993) "Implementing an Insertion Heuristic for Vehicle Routing on Parallel Hardware", *Computers and Operations Research* 20, 737-745
80. **Fredman M.L., Johnson D.S., McGeoch L.A. y Ostheimer G.** (1995), "Data Structures for Traveling Salesmen", *J. Algorithms*, 18, 432-479
81. **Freisleben B. y Merz P.** (1996) "A Genetic Local Search Algorithm for Solving Symetric and Asymetric Traveling Salesman Problmes*" Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, Nagoya, Japan, pp. 616-621
82. **Gambardella L.M., Taillard E. y Agazzi G.,** (1999), "MACS-VRPTW: A Multiple Ant Colony Systema for Vehicle Routing Problems with Time Windows", en *New Ideas in Optimization*, D. Corne, M. Dorigo y F. Glover (eds), 63-76, McGraw-Hill, London
83. **Garcia B.L., Potvin J.Y. Y Rousseau J.M.**, (1994), "A Parallel Implementation of the Tabu Search Heuristic for Vehicle Routing Problems with Time Window Constraints", *Comp. Oper. Res*, Vol. 21, No. 9, p. 1025-1033.
84. **Gehring H. y Homberger J.**, (1999), "A Parallel Hybrid Evolutionary Meta-heuristic for the Vehicle Routing Problem with Time Windows", en *Proceedings of EUROGEN99 - Short Course on Evolutionary Algorithms in Engineering and Computer Science*, Reports of the Department of Mathematical Information Technology, Series A. Collections, No. A 2/1999, K. Miettinen, M. Mäkelä y J. Toivanen (eds), 57-64, University of Jyväskylä, Finland
85. **Gehring H. y Homberger J.**, (2001), "Parallelization of a Two-Phase Metaheuristic for Routing Problems with Time Windows", *Asia-Pacific Journal of Operations Research* 18, 35-47
86. **Gendreau M. y Laporte G.**, (1995), "Issue on: Freight Transportation", in *Ann. Oper. Res*., Vol. 61.
87. **Gendreau M., Laporte G. y Potvin J.Y.**, (1994), *"Local Search Algorithms for the Vehicle Routing Problem"*, working paper, CRT-963, Centre de Recherche sur les Transport, Universite de Montreal.
88. **Gendreau M., Laporte G. y Seguin R.**, (1994), *"A Tabu Search Heuristic for the Vehicle Routing Problem with Stochastic Demands and Customers"*, working paper, CRT-987, Centre de Recherche sur les Transport, Universite de Montreal.
89. **Gendreau M.,Hertz A. y Laporte G.**, (1992), "New Insertion and Postoptimization Procedures for The Traveling Salesman", *Oper. Res*., Vol. 40, No. 6, p. 1086-1094.
90. **Gendreau M., Hertz A. y Laporte G.**, (1994), "A Tabu Search Heuristic for the Vehicle Routing Problem", *Manag. Sci.,* Vol. 40, No. 10, p. 1276-1290.
91. **Gendreau M., Laporte G. y Seguin R.**, (1995), An Exact algorithm for the vehicle routing problem with stochastic customers and demands", *Transp. Sci.* Vol. 29, p. 143-155.
92. **Gendreau M., Laporte G. y Seguin R.**, (1996), "Stochastic vehicle routing", *Eur. J. Oper. Res*. Vol. 88, p. 3-12.
93. **Gendreau M., Hertz A., Laporte G. y Stan M.,** (1998), "A Generalized Insertion Heuristic for the Traveling Salesman Problem with Time Windows", *Operations Research* 46, 330-335
94. **Gillet B. y Miller L.R.**, (1974), "A Heuristic Algorithm for the Vehicle Dispatch Problem", *Oper. Res*., Vol. 22, p. 340-349.

95. **Glover F**., (1986), "Future Paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research* 13, 533-549
96. **Glover F.,** (1989) "Tabu Search-Part I", *Journal on Computing* 1, 190-206
97. **Glover F.**, (1991), "*Multilevel Tabu Search and Embedded Search Neighborhoods for the Traveling Salesman Problem*", working paper, College of Business Administration, University of Colorado, Boulder.
98. **Glover F.**, (1992), "New Ejection Chain and Alternating Path Methods for Traveling Salesman Problems", en *Computer Science and Operations Research: New Developments in Their Interfaces*, O. Balci, R. Sharda, y S. Zenios (eds), 449-509, Pergamon Press, Oxford
99. **Goldberg D.,** (1989), "*Genetic Algorithms in Search, Optimization, and Machine Learning*", Addison Wesley Publishing Company Inc., New York
100. **Golden B.L. y Assad A.A.**, (1986), "Special issue on: Vehicle Routing with Time-Window Constraints: Algorithmic Solutions", *Am. J. Math. Mgmnt. Sci*, Vol. 6.
101. **Golden B.L. y Assad A.A.**, (1986), "Perspectives on Vehicle Routing: Exciting New Developments", *Oper. Res*., Vol. 34. No. 5, p. 803-810.
102. **Golden B.L. y Assad A.A.** (eds.), (1988), *Vehicle Routing: Methods and Studies*, Studies in Manag. Science and Systems 16, North-Holland Publ., Amsterdam, 1988.
103. **Golden B. y Stewart W.R.,** (1985), "Empirical Analysys of Heuristics" en *The Traveling Salesman Problem*, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan y D.B. Shmoys (eds), 207-249, John Wiley & Sons, Chichester
104. **Golden B. L. y Wasil E.A**., (1987), *"*Computerized Vehicle Routing in the Soft Drink Industry*"*, *Oper. Res*., Vol. 35, No. 1, p. 6-17.
105. **Gorges-Schleuter M.** (1991), "*Genetic Algorithms and Population Structures. A Massively Parallel Algorithm*", PhD. Thesis. University of Dortmund
106. **Hall, R.W.,** (1999) *Handbook of Transportation Science* Ed. Kluwer
107. **Hall R.W.**, (2002), "Vehicle Routing Software Survey", *OR/MS Today*, INFORMS, August 2002
108. **Hamacher A., y Moll C.**, (1996), "A New Heuristic for Vehicle Routing with Narrow Time Windows", en *Operations Research Proceedings 1996*, Selecet papers of the symposium (SOR'96), Braunschweig, Germany, September 3-6, U. Derigs, W. Gaul, R.H. Möhring y K.P. Schuster (eds), 301-306, Springer Verlag, New York.
109. **Helbig-Hansen K.H., y Krarup J.** (1974) "Improvements of the Held-Karp Algorithm for the Symetric Traveling Salesman Problem". *Math. Programming*, 7, 87-96
110. **Held M. y Karp R.**, (1970) " The Traveling Salesman Problem and Minimun Spanning Trees", *Ops Res.* 18, 1138-1162
111. **Held M. y Karp R.**, (1971) "The Traveling Salesman Problem and Minimun Spanning Trees Part II", *Math Prog*. 1, 6-25
112. **Held M., Wolfe P., y Crowder H.**, (1974), "Validation for Subgradient Optimization", *Math. Prog*. 6, 62-88
113. **Homberger J. y Gehring H.**, (1999), "Two Evolutionary Meta-heuristics for the Vehicle Routing Problem with Time Windows", *INFORMS Journal on Computing* 37, 297-318
114. **Hong S.C. y Park Y.B.**, (1999), "A Heuristic for Bi-Objective Vehicle Routing woth Windows Constraints", *International Journal of Production Economics* 62, 249-258
115. **Hooker, J.N.**, (1995), " Testing Heuristics: We Have It All Wrong", *Journal of Heuristics* 1, 33-42
116. **Ioannou G., Kritikos M., y Prastacos G.**, (2001), "A Greedy Look-Ahead Heuristic for the Vehicle Routing Problem with Time Windows", *Journal of the Operational Research Society* 52, 523-537

117. **Jaffar J., y Lassez J.L.**, (1986), "*Constraint Logic Programming*", Technical Report 86/73, Department of Computer Science, Monash University, Australia
118. **Jaffar J., y Maher M.J.**, (1994), "Constraint Logic Prgramming: A Survey", *Journal of Logic Programming* 19/29, 503-581
119. **Johnson D.S.**, (1990), "Local Optimization and the Traveling Salesman Problem", en *Proceedings 17th International Colloquium on Automata Languages and Programming*, 446-460
120. **Johnson D.S., McGeoch L.A. y Rothberg E.E.**, (1996), "Asymptotic Experimental Analysis for the Held-Karp Traveling Salesman Bound" *Proc. 7th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pp. 341-350.
121. **Karp R.** (1972), *Reductibility among combinatorial problems. Complexity of Computer Computations* (Editado por R. Miller y J. Thatcher) pp. 85-104. Plenum Press, Nueva York
122. **Kelly J.P. y Xu J.** (1997) "*A Generic Set-Covering-Based Heuristic for Vehicle Routing Problems*". Working Paper
123. **Kelly J.P. y Xu J.** (1999) "A Set-Partitioning Heuristic for the Vehicle Routing Problem", *INFORMS Journ. On Comp.* 11, 161-172
124. **Kilby P., Prosser P, y Shaw P.,** (1999) "Guided Local Search for the Vehicle Routing Problem with Time Windows", en *META-HEURISTICS Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I.H. Osman y C. Roucairol (eds), 473-486, Kluwer Academic Publishers, Boston
125. **Kirkpatrick S., Gelatt C.D., y Vecchi P.M.** (1983), "Optimization by Simulated Annealing", *Science* 220, 671-680.
126. **Knight K., y Hofer J.**, (1968), "Vehicle Scheduling with Timed and Connected Calls: A Case Study", *Operational Research Quarterly* 19, 299-310
127. **Knox J.,** (1994), "Tabu Search Performance on the Symmetric Traveling Salesman Problem", *Computers Ops Res*., Vol. 21, No. 8, pp. 867-876,
128. **Kontoravdis G., y Bard J.F.**, (1995), "A GRASP for the Vehicle Routing Problem with Time Windows", *INFORMS J. Comp*., Vol. 7, No. 1, p. 10-23.
129. **Korte B.**, (1988) "*Applications of Combinatorial Optimization*", Technical Report 88541-OR Institute fur Okonometrie und Operations Research, Bonn, Alemania
130. **Laguna M. y Moscato P.,** (1995) "Algoritmos Genéticos" Capítulo 3, del libro "*Optimización Heurística y Redes Neuronales*", B. A. Diaz, Ed. Paraninfo, Madrid, España, (1996).
131. **Landeghem H.R.G.**, (1988), "A bi-criteria heuristic for the vehicle routing problem with time windows", *Eur. J. Oper. Res*., Vol. 36, p. 217-226.
132. **Larsen J**., (1999), "*Parallelization of the Vehicle Routing Problem with Time Windows*", PhD. Thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark
133. **Lau H.C. Lim Y.F. y Liu Q.**, (2000), "*Diversification of Neighborhood via Constraint-Based Local Search and its Applications to VRPTW*", Working Paper, School of Computing, National University of Singapore
134. **Le Bouthillier A., Crainic T.G. y Keller R.K.**, (2001), "*Co-operative Parallel Method for Vehicle Routing Problems with Time Windows*", Presentado en MIC'2001, 4th Metaheuristics International Conference, July 16-20, Porto, Portugal
135. **Li H., Lim A. y Huang J.**, (2001), "*Local Search with Annealing -like Restarts to solve the VRPTW*", working paper, Department of Computer Science, National University of Singapore
136. **Lin S.**, (1966), "Computer Solutions of the Traveling Salesman Problem", *Bell System Tech. J*., Vol. 44, p. 2245-2269.

137. **Lin S. y Kernighan B.W.**, (1973), "An effective heuristic algorithm for the Traveling Salesman Problem", *Oper. Res*., Vol. 21, p. 2245-2269.
138. **Liu F.H. y Shen S-Y.**, (1999), "A Route-Neighborhood-Based Metaheuristic for the Vehicle Routing Problem with Time Windows", *European Journal of Operations Research* 118, 485-504
139. **Madsen O.B.G.**, (1976), "Optimal Scheduling of trucks-a Routing Problem with Tight Due Times of Delivery" en *Optimization Applied to Transportation Systems*, H. Stroble, R. Genser y M. Etschmaier (eds), 126-136, International Institte for Applied System Analysis, Laxenburgh
140. **Malek M., Guruswamy M. y Pandya M.**, (1989), "Serial and Parallel Simulated Anealling and Tabu Search Algorithms for the Traveling Salesman Problem", *Ann. Oper. Res*., Vol. 21, p. 59-84.
141. **Martin O., Otto S.W., y Felten E.W.**, (1991), "Large Step Markov Chains for the Traveling Salesman Problem", *Complex Systems* 5 (3), 299-326
142. **Martin O., Otto S.W. y Felten E.W.**, (1992), "Large Step Markov Chains for the Traveling Salesman Problem Incorporationg Local Search Heuristics", *Operations Research Letters* 11 (4), 219-224
143. **Martin O. y Otto S.W.**, (1994), "Combining Simulated Annealing with Local Search Heuristics", en *Metaheuristics in Combinatoric Optimization* G Laporte y I. Osman (eds).
144. **Metropolis W., Rosenbluth A., Rosenbluth M., Teller A. y Teller E.**, (1953), "Equation of the State Calculations by Fast Computing Machines", *Journal of Chemical Physics* 21, 1087-1092
145. **Mikkola T.**, (2000), "*Algorithm Library for Large Scale Vehicle Routing*", Master thesis, Department of Computer Science and Technology, Helsinki University of Technology, Espoo, Finland
146. **Mingozzi A., Giorgi S. y Baldacci R.** (1999), "An Exact Method for the Vehicle Routing with Backhauls", *Transport. Science*, 33, 315-329
147. **Mladenovic N. y Hansen P.**, (1997), "Variable Neighborhood Search", *Computers & Operations Research* 24, 1097-1100
148. **Moscato P. y Norman M.G.** (1992) "A 'Memetic' Approach for the Traveling Salesman Problem. Implementations of a Computational Ecology for Combinatorial Optimization on Message-Passing Systems", *Parallel Computing and Transputer Applications* , ed. por M. Valero, E. Onate, M. Jane, J.L. Larriba and B. Suarez, Ed. IOS Press, Amsterdam, pp. 187-194
149. **Mühlenbein H., Gorges Schleuter M. y Kramer O.**, (1988), "Evolution Algorithm in Combinatorial Optimization", *Parallel Computing* 21, 65-85
150. **Or I.**, 1976, "*Traveling Salesman-Type Combinatorial Problems and their Relation to the Logistics of Regional Blood Banking*", PhD thesis, Northwstern University, Evanston, Illinois
151. **Osman I.H.**, (1993), "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem", *Ann. Oper. Res.,* Vol. 41, p. 421-451.
152. **Paessens, H.**, (1988), "The Savings Algorithm for the Vehicle Routing Problem", *Eur. J. Oper. Res.*, Vol. 34, p. 336-344.
153. **Potvin J.Y.**, (1993), "The Traveling Salesman problem: A Neural Network Perspective", *INFORMS J. Comp*., Vol. 5, No. 4, p. 328-348.
154. **Potvin, J.Y. y Guertin F.**, (1996), "The Clustered Traveling Salesman Problem: A Genetic Approach", en *Meta-Heuristics: Theory & Applications*, Eds.: I. H. Osman & J. P. Kelly, Kluwer Academic Publ. Boston, p. 619-632.

155. **Potvin J.Y. y Robillard C.**, (1995), "Integrating Operations Research and Neural Networks for Vehicle Routing", en: *The Impact of Emerging Technologies on Computer Science and Operations Research*, Stephen G. Nash & Ariela Sofer (Eds.), p. 245-262.
156. **Potvin J.Y. y Rousseau J.M.**, (1993), "A Parallel Route Building Algorithm for the Vehicle Routing and Scheduling Problem with Time Windows", *Eur. J. Oper. Res*., Vol. 66, p. 331-340.
157. **Potvin J.Y. y Rousseau J.M.**, (1995), "An Exchange Heuristic for Routeing Problems with Time Windows", *J. Oper. Res. Soc*., Vol. 46, p. 1433-1446.
158. **Potvin J.Y., Dufors G. Rousseau J.M.**, (1993), "Learning Vehicle Dispatching with Linear Programming Models", *Comp. Ops. Res*., Vol. 20, p. 371-380.
159. **Potvin J.Y., Lapalme G. y Rousseau J.M.**, (1989), "A generalized K-Opt Exchange Procedure for the MTSP", *INFOR*, Vol. 27, no. 4, p. 474-481.
160. **Potvin J.Y., Shen Y. y Rousseau J.M.**, (1992), "Neural Networks for Automated Vehicle Dispatching", *Comp. Ops. Res*., Vol. 19, No. 3/4, p. 267-276.
161. **Potvin J.Y., Kervahut T., Garcia B.L. y Rousseau J.M.**, (1996), "The Vehicle Routing Problem with Time Windows - Part 1: Tabu Search", *INFORMS J. Comp*., Vol. 8, No. 2, p. 158-164.
162. **Potvin J.Y., y Bengio S.**, (1996), "The Vehicle Routing Problem with Time Windows - Part 2: Genetic Search", *INFORMS J. Comp*., Vol. 8, No. 2, p. 165-172.
163. **Potvin J.Y., Dubé D. y Robillard C.**, (1996), "A Hybrid Approach to Vehicle Routing Using Neural Networks and Genetic Algorithms", *Applied Intelligence* 6, 241-252
164. **Prosser P. y Shaw P.**, (1996), "*Study of Greedy Search with multiple improvements heuristics for Vehicle Routing Problems*", working paper, University of Strathclyde, Glasgow, Scotland
165. **Pullen H., y Webb M.**, (1967), "A Computer Application to a Transport Scheduling Problem", *Computer Journal* 10, 10-13
166. **Rahoual M., Kitoun B., Mabed M.H., Bachelet V. y Benameur F.**, (2001), "*Multicriteria Genetic Algorithm for the Vehicle Routing Problem with Time Windows*", Presentado en la MIC'2001, 4th Metaheuristic International Conference, July 16-20, Porto, Portugal
167. **Rego C. y Roucairol C.**, (1994), *" An Efficient Implementation of Ejection Chain Procedures for the Vehicle Routing Problem"*, working paper, PRISM Lab., University of Versailles, France.
168. **Rego C. y Roucairol C.**, (1995), "Using Tabu Search for solving a Dynamic Multi-Terminal Truck Dispatching Problem", *Eur. J. Oper. Res*., Vol. 83, p. 411-429.
169. **Rego C. y Roucairol C.**, (1996), " A Parallel Tabu Search Algorithm Using Ejection Chains for the Vehicle Routing Problem", en *Meta-Heuristics: Theory & Applications*, Eds.: I. H. Osman & J. P. Kelly, Kluwer Academic Publ. Boston, p. 661-676.
170. **Rego C**, (1997) "*Relaxed Tours and Path Ejections for the Traveling Salesman Problem"* Working Paper Universidade Portucalense Depto. Informatica, Porto, Portugal
171. **Rego C.**, (1998), "A Subpath Ejection Method for the Vehicle Routing Problem", *Management Science* 44, 1447-1459
172. **Reinelt G.**, (1992), "TSPLIB-A traveling Salesman Problem Library", *ORSA Journal on Computing* 3, 376-384
173. **Reinelt G.**, (1995), *"TSPLIB 95"*, working paper, Institut fuer Angewandte Mathematik, Universitaet Heidelberg.
174. **Renaud J., Boctor F. y Laporte G**, (1996), "An Improved Petal Heuristic for the Vehicle Routing Problem", *Eur. Jour. of Ops. Res. Soc.* 17, 329-326

175. **Riise A. y Stolevik M.**, (1999), "*Implementation of Guided Local Search for the Vehicle Routing Problem*", SINTEF Internal Report STF42 A99013, SINTEF Applied Mathematics, Norway
176. **Rochat Y. y Taillard E.**, (1995), "Probabilistic Diversification and Intensification in Local Search for Vehicle Routing", *Journal of Heuristics* 1, 147-167
177. **Rohe A.** (1995) *Parallel Lower and Upper Bounds for Large TSPs* PhD Thesis, ZAMM Volume 77, Suplement 2, pp 429-432, 1997
178. **Russel R.A.**, (1977), "!An Effective Heuristic for the M-Tour Traveling Salesman Problem with Some Side Conditions", *Oper. Res*., Vol 25, No. 3, p. 517-524.
179. **Russel R.A.**, (1995), "Hybrid Heuristics for the Vehicle Routing Problem with Time Windows", *Transp. Sci*., Vol. 29, No. 2, p. 156-166.
180. **Savelsbergh M.W.P.**, (1986), "Local Search in Routing Problems With Time Windows", *Ann. Oper. Res.*, Vol. 4, p. 285-305.
181. **Savelsbergh M.W.P.**, (1990), "An Efficient Implementation of Local Search Algorithms for Constrained Routing Problems", *Eur. J. Oper. Res*., Vol. 47, p. 75-85.
182. **Savelsbergh M.W.P.**, (1992), "The Vehicle Routing Problem with Time Windows: Minimizing Route Duration", *INFORMS J. Comp*., Vol. 4, No. 2, p. 146-154.
183. **Schulze J. y Fahle T.**, (1999), "A Parallel Algorithm for the Vehicle Routing Problem with Time Windows Constraints", *Annals of Operations Research* 86, 585-607
184. **Shaw P.**, (1997), "*A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems*", working paper, Department of Computer Sciences, University of Strathclyde, Glasgow, Scotland
185. **Shaw P.**, (1998), "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems" en *Principles and Practice of Constraint Programming - CP98, Lecture Notes in Computer Science*, M. Maher y J.F. Puget (eds), 417-431, Springer-Verlag, New York.
186. **Solomon, M.**, (1983), *"Vehicle Routing and Scheduling With Time Window Constraints: Models and Algorithms"*, working paper, No. 83-42, College of Business Administration, Northeastern Univeristy.
187. **Solomon M.**, (1986), "On the Worst-Case Performance of Some Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constarints", *Networks*, Vol. 16, p. 161-174.
188. **Solomon M.**, (1987), "Algorithms for the vehicle routing and scheduling problem with time window constraints", *Oper. Res*. Vol. 32, p.254-265.
189. **Solomon M., Baker E.K. y Schaffer J.R.**, (1988), "Vehicle Routing and Scheduling Problems with Time Windows Constraints: Efficient Implementations of Solution Improvement Procedures", en: *Vehicle Routing: Methods and Studies*, Ed. Golden B. G. and Assad A. A., Elsevier Science Publishers B.V. (North Holland), p. 85-105.
190. **Solomon M. y Desrosiers J.**, (1988), "Time Window Constrained Routing and Scheduling Problems", *Transp. Sci*., Vol. 22, No. 1, p.1-13.
191. **Stewart W. y Golden B.** (1979) "A Vehicle Routing algorithm based on Generalized Lagrange Multipliers " *Procc. Of AIDS Annual Convention*, Ed. L. Moore, K. Monroe, B. Taylor, pp 108-110, Vol. 2 New Orleans USA
192. **Suh J.Y. y Van Gucht D.** (1987)"*Incorporating Heuristic Information into Genetic Search*", ICGA 1987, 170-176
193. **Taillard E., Laporte G. y Gendreau M.**, (1995), *"Vehicle Routing with Multiple Use of Vehicles"*, working paper, CRT-95-19, Centre de Recherche sur les Transport, Univerity of Montreal.
194. **Taillard E.**, (1993), "Parallel Iterative Search Methods for Vehicle Routing Problems", *Networks*, Vol. 23, p. 661-673.

195. **Taillard E.**, (1995), *"A Heuristic Column Generation Method for the Heterogeneous Fleet VRP"*, working paper, Centre de Recherche sur les Transport, Univerity of Montreal.
196. **Taillard E. y Rochat Y.**, (1995), *"Probabilistic Diversifications and Intensifications in Local Search for Vehicle Routing"*, working paper, CRT-95-13, Centre de Recherche sur les Transport, Univerity of Montreal, Forthcoming in Journal of Heuristics.
197. **Taillard, E., Badeau P., Gendreau M., Francois Guertin F. y Potvin J.Y.**, (1995), *"A New Neighborhood Structure for the Vehicle Routing Problem with Time Windows"*, working paper, Centre de Recherche sur les Transport, Univerity of Montreal.
198. **Taillard, E., Badeau P., Gendreau M., Francois Guertin F. y Potvin J.Y.**, (1997), "A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows" *Transportation Science* 31, 170-186
199. **Taillard E.**, (2001), "*Comparison of Non-Deterministic Iterative Methods*", Presentado al MIC'2001, 4º Metaheuristics International Conference, Julio 16-20, Porto, Portugal
200. **Tan K.C., Lee L.H. y Zhu K.Q.**, (2000), "Heuristic Methods for the Vehicle Routing Problem with Time Windows", en *Proceedings of the 6th International Symposium on Artificial Intelligence & Mathematics*, Ft. Lauderdale, Florida
201. **Thangiah S.R, Osman I. Y Sun T.**, (1994), "*Hybrid Genetic Algorithm, Simulated Annealing and Tabu Search Methods for the Vehicle Routing Problems with Time Windows*", Working paper UKC/IMS/OR94/4, Institute of Mathematics and Statistics, University of Kent, Canterbury
202. **Thangiah S.R.**, (1995a), *"Vehicle Routing with Time Windows Using Genetic Algorithms"*, working paper, Comp. Sci. Dept., Slippery Rock University, Submitted to the book: "Applications Handbook of Genetic Algorithms: New Frontiers".
203. **Thangiah S.R.**, (1995b), "An Adaptive Clustering Method Using a Geometric Shape for Vehile Routing Problems with Time Windows", en *Proceedins of the 6th International Conference on Genetic Algorithms*, L.J. Eshelman (ed), 536-543, Morgan Kauffmann, San Francisco
204. **Thangiah S.R. y Chang H**., (1996), *"PHOENIX: A Graphical User Interface System for School Bus Routing"*, working paper, Comp. Sci. Dept., Slippery Rock University.
205. **Thangiah, S.R., Osman I.H. y Sun T.**, (1995), *"Metaheuristics for Vehicle Routing Problems with Time Windows"*, working paper, Comp. Sci. Dept., Slippery Rock University.
206. **Thangiah S.R., Osman I.H. y Sun T.**, (1996), *"Hybrid Genetic Algorithm, Simulated Annealing and Tabu Search Methods for Vehicle Routing Problems with Time Windows"*, working paper, Comp. Sci. Dept., Slippery Rock University.
207. **Thangiah S.R., Potvin J.Y. y Sun T.**, (1995), *"Heuristic Approaches to Vehicle Routing with Backhauls and Time Windows"*, working paper, Comp. Sci. Dept., Slippery Rock University, Submitted to Comp. & Oper. Res..
208. **Thangiah S.R.,. Osman I.H., Vinayagamoorthy R. y Sun T.**, (1995), "Algorithms for the Vehicle Routing Problems with Time Deadlines", *Am. J. Math. Mgmnt. Sci*., Vol 13 (3&4), p. 323-355.
209. **Thompson P.M. y Psaraftis H.N.**, (1993), "Cyclic Transfer Algorithms for Multivehicle Routing and Scheduling Problems", *Oper. Res*., Vol. 41, No. 5, p. 935-946.
210. **Toth P. y Vigo D.**, (1996), "Fast Local Search Algorithms for the Handicapped Persons Transportation Problem", in *Meta-Heuristics: Theory & Applications*, Eds.: I. H. Osman & J. P. Kelly, Kluwer Academic Publ. Boston, p. 677-690.
211. **Toth P., y Vigo D.,** (2002) *The Vehicle Routing Problem* SIAM Monographs

212. **Troyon M.** (1988), *Quelques Heuristiques et Resultats Asymptotiques pour Trois Problems d'Optimisation Combinatoire*., Ph.D. Thesis, Ecole Polytechnique Federale de Lausanne
213. **Vliet A, Guus C., Boender E., y Rinnooy Kan A. H. G.** (1992) "Interactive optimization of bulk sugar deliveries" *Interfaces* 22:3 pp:4-14
214. **Vouduris C.**, (1997), "*Guided Local Search for Combinatorial Problems*", PhD. Thesis, Department of Computer Science, University of Essex, Colchester, UK
215. **Vouduris C. y Tsang E.**, (1998), "Guided Local Search", *European Journal of Operations Research*, 113, 80-119
216. **Wee Kit H., Chin J. y Lim A**., (2001), "A Hybrid Search Algorithm for the Vehicle Routing Problem with Time Windows", *International Journal on Artificial Intelligence Tools*.
217. **Weigel D. y Cao B.** (1999) "Applying GIS and OR Techniques to Solve Sears Technician-Dispatching and Home-Delivery Problems" *Interfaces* 29:1, 114
218. **Xu, J., y Kelly P.**, (1995), *"A Robust Network Flow-Based Tabu Search Approach for the Vehicle Routing Problem"*, working paper, Graduate School of Business, University of Colorado at Boulder.
219. **Zachariasen M. y Dam M.** (1995), "*Tabu Search on the Geometric Traveling Salesman Problem*", Presentado en Metaheuristics Internacional Conference 1995
220. **Zhu K.Q.**, (2000), "*A New Genetic Algorithm for VRPTW*", presentado en IC-AI 2000, Las Vegas, U.S.A.

Anexo I

Encuesta sobre Software

Fuente: Hall (2002)

Producto	Proveedor	Año	Tipos de flota que actualmente utilizan este tipo de software								Información Comercial				
			Transporte mercancías	Recogida y Entrega	Cargas Parciales	Cargas totales	Paquetería	Buses	Taxis	Programación de equipos de servicio	Una licencia (50 rutas)	¿Se incluye mapa de la región?	Proveedor del Mapa	Coste de hora del instalador	Nº de Horas instalación
ArcLogistics Route 3	ESRI	1999	sí	sí	sí	-	sí	-	-	sí	$12,000	sí	AGDT DynaMap	$150	8 horas
Compass	RouteLogic	1999	-	sí	-	-	-	sí	-	-	-	-	-	-	-
Direct Route	Appian Logistics Software, Inc.	1996	sí	sí	sí	-	sí	-	-	sí	$14,375	sí	GDT	$150	16 horas
Edgar Transp. Management System	Edgar Management Consultants, Inc.	1972	sí	sí	-	-	sí	sí	-	-	Contacte	sí	-	incluido	incluido
Fleetwise	Descartes Systems Group	1987	sí	sí	sí	-	sí	-	-	sí	Contacte	-	-	-	-
ILOG Dispatcher	ILOG	1997	sí	sí	sí	sí	sí	sí	sí	sí	Contacte	-	-	contacte	-
Intertour/Interload	Intertour/Interload	1983	sí	sí	sí	sí	sí	sí	sí	sí	Contacte	sí	NavTech, AND, etc.	$125	2-5 días
Manugistics Fleet Management	Manugistics, Inc.	1985	sí	sí	sí	sí	-	-	-	-	-	sí	MapInfo TAB	$285	-
Optrak4	Optrak Distribution Software	2001	sí	sí	sí	sí	-	-	-	-	-	sí	contacte	contacte	contacte
Prophesy Mileage & Routing	Prophesy Transportation Solutions, Inc.	1984	sí	-	-	-	-	sí	-	sí	Contacte	-	-	-	-
Protour	Prologos Planung und Beratung	1994	sí	sí	-	-	sí	-	-	sí	$18,000	sí	NavTech Europe	$100	10 horas
RoadNet 5000	UPS Logistics Group	1983	sí	sí	sí	-	-	-	-	-	$60,500	-	GDT, NavTech	$118.75	72 horas
Roadshow System	Descartes Systems Group	1987	sí	sí	sí	-	sí	-	-	sí	contacte	-	-	-	-
RoutePro	CAPS Logistics	1997	sí	sí	sí	sí	-	-	-	-	$30,00+	sí	contacte	contacte	contacte
RouteSmart	RouteSmart Technologies, Inc.	1989	-	sí	-	-	-	-	-	sí	contacte	-	-	-	contacte
Routronic 2000	Carrier Logistics, Inc.	1985	sí	sí	sí	-	sí	-	-	-	$30,000	-	Mapinfo-Option	$100	50-80 horas
SHORTREC product suite	ORTEC	1981	sí	sí	sí	sí	sí	sí	sí	sí	contacte	sí	NavTech, TeleAtlas, etc.	contact	contacte
STARS	SAITECH, Inc.	1995	sí	sí	sí	sí	sí	-	-	sí	contacte	-	Chicago Map, MapInfo	$150	2-10 días
Territory Planner	UPS Logistics Group	1983	sí	sí	sí	-	-	-	-	-	$73,500	-	GDT, NavTech	$118.75	72 horas
tmsRouter	GeoComtms	2000	sí	sí	-	-	sí	-	-	sí	contacte	sí	-	contacte	contacte
tmsZoneDesigner	GeoComtms	2001	sí	sí	-	-	sí	-	-	sí	contacte	sí	contacte	contacte	contacte
Trapeze	Trapeze Software Group, Inc.	1992	-	-	-	-	-	sí	-	-	contacte	-	-	-	-
TruckSTOPS Routing & Scheduling for Windows	Micro Analytics, Inc.	1984	sí	sí	sí	sí	sí	-	-	sí	$9,500	-	PC-Miler, etc.	-	contacte
VersaTrans Software	VersaTrans Solutions, Inc.	1982	-	-	-	-	-	sí	-	-	-	sí	NavTech	contacte	10 horas

Producto	Otras características	Nºaprox clientes y % clientes privados	Clientes más importantes
ArcLogistics Route 3	-	sobre 2,500/50%	-
Compass	-	30+/2%	American Media (privada) & Florida Rural ITS Project (administración)
Direct Route	Optimización de recursos	175/65%	Walgreens, JB Hunt, Scheider, Transervice, Nabisco
Edgar Transportation Management System	Sistema adaptable a las necesidades de cada cliente	20 empresas/10%	Fort Worth ISD, Los Angeles County Office of Education
Fleetwise	Planificación de rutas, envíos y comunicaciones inalámbricas. Adaptabilidad máxima a las necesidades del cliente	1,500 delegaciones	Coca-Cola, Ambev, FedEx, Air Ground Freight Service Division
ILOG Dispatcher	Cargas y descargas modificables, compartimentos de vehículos modificables, planificación de rutas o nodos a escoger	300+	-
Intertour/Interload	Optimización de cargas, planificación estratégica, transportes intermodales	500	Swiss Post, Unilever, Daimler Chrysler, Volkswagen, BMW, Rockwool, Auchan, Feldschloesschen
Manugistics Fleet Management	Horas punta, calles de un solo sentido, partición de cargas, frecuencias de envíos, desarrollo en esquema de subastas	1,000 instalaciones	Advance Auto, Ahold, Albertsons, Baxter Healthcare, Fleming, Kroger, Nash Finch, Perseco, Winn Dixie, Safeway, McLane
Optrak4	Adaptable a través de scripts, optimización por fases,	50	-
Prophesy Mileage & Routing	Mapas a nivel de callejero	25,000 instalaciones	Empresas del grupo de Fortune 100, y pequeñas empresas
Protour	Modulo de optimización territorial	20/90%	Direct Parcel Service, Germany
RoadNet 5000	-	849/99%	Pepsi, Sysco, U.S. Foods
Roadshow System	Planificación integral de rutas, comunicaciones inhalámbricas	1,500 delegaciones	Coca-Cola, Ambev, FedEx, Air Ground Freight Service Division
RoutePro	Multicompartimentación, equilibrado de cargas.	100+/75%	Daimler Chrysler, Ford Motor, Excel Logistics, GA-Pacific, Fresh Express
RouteSmart	Diseñado para reparto de periódicos, correo postal, y recogida de basuras	100+/50%	FedEx home Delivery USPS, New York Times, Coned, 50+ municipalities
Routronic 2000	Paquete total, con integración de tarifas y precios	90	Toll Holomes, Australia, Conway Transe Services U.S.
SHORTREC product suite	Integrable con otras aplicaciones como gestión de almacenes.	100/80%	BP, Texaco, TNT Logistics, Lafarge, Yellow Freight System
STARS	Cercanías y largas distancias, múltiples depósitos, rutas interregionales, y programación de servicios.	20/70%	Nisseki-Mitsubishi Oil Company (Japan)
Territory Planner	-	849/99%	Pepsi, Sysco, Anhesuser-Busch
tmsRouter	Algoritmo genético de optimización, arquitectura cliente-servidor, TCP/IP networking	Contacte	Carroll Independent Fuel, Southern Maryland Oil, Purolator, Multi-Marques
tmsZoneDesigner	Algoritmo genético de optimización, arquitectura cliente-servidor, TCP/IP networking	Contacte	Carroll Independent Fuel, Southern Maryland Oil, Purolator, Multi-Marques
Trapeze	Programación dinámica de demandas, rutas, fijas, o rutas flexibles.	800 productos instalados	Dallas Area rapid Transit, Capital Metropolitan Trans. Authority (Austin, TX), Spokane Transit, Van Tran (Tucson, AZ)
TruckSTOPS Routing & Scheduling	Planificación de almacenes, cargas y descargas simultánea, optimización para múltiples días, varios almacenes.	2,400 sistemas vendidos	Rollins, Home Depot, Simmons Mattress, UPS Worldwide, 7-Eleven, Dominoes, Nestle, Pepsi AMerica, Miller Brewing, Chicago Tribune
VersaTrans Routing & Planning Software	Paquete de relocalización incluyendo creación automática de rutas.	700	Buffalo, NY; Davis, CO

Producto	Tamaño máximo del problema			Funciones de planificación de rutas					Características				
	Paradas	Vehículos	Almacenes	Por nodos	Rutas diarias	Planificación de rutas y análisis	Rutas en tiempo real	Por arcos	Asigna conductores individuales	Instrucciones detalladas de conducción minuto a minuto	Previsión automática de envíos	Manifiestos de carga	Plan de cargas para cada vehículo
ArcLogistics Route 3	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	-	sí	sí	-	sí	sí
Compass	10	1	100	-	sí	sí	sí	-	sí	sí	sí	sí	sí
Direct Route	16	1	500	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
Edgar Transportation Management System	ilimitado	ilimitado	ilimitado	sí	sí	sí	-	-	sí	sí	-	sí	sí
Fleetwise	32,000+	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
ILOG Dispatcher	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	-	-	-	-
Intertour/Interload	850/8,000	500/8,000	99/2,000	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
Manugistics Fleet Management	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	-	sí	sí	sí	sí	sí
Optrak4	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
Prophesy Mileage & Routing	ilimitado	ilimitado	ilimitado	-	-	-	-	-	-	-	-	-	-
Protour	4	1	ilimitado	sí	sí	sí	-	-	-	sí	-	sí	-
RoadNet 5000	ilimitado	ilimitado	ilimitado	-	sí	-	-	-	sí	sí	-	-	-
Roadshow System	32,000+	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
RoutePro	-	-	-	-	sí	sí	sí	-	sí	sí	-	sí	-
RouteSmart	ilimitado	ilimitado	ilimitado	sí	sí	sí	-	sí	-	-	-	-	-
Routronic 2000	ilimitado	ilimitado	ilimitado	-	sí	sí	sí	-	sí	-	-	sí	sí
SHORTREC product suite	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
STARS	100	1	3,006	sí	sí	sí	sí	sí	sí	sí	-	-	sí
Territory Planner	ilimitado	ilimitado	ilimitado	-	-	sí	-	-	sí	sí	-	-	-
tmsRouter	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	-	sí	sí	-	sí	sí
TmsZoneDesigner	ilimitado	ilimitado	ilimitado	sí	sí	sí	-	-	sí	sí	-	sí	sí
Trapeze	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	sí	sí	-
TruckSTOPS	ilimitado	ilimitado	ilimitado	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
VersaTrans	500	3	ilimitado	sí	sí	sí	-	-	sí	sí	-	-	-

Producto	Algoritmo considerado								Acepta	
	Programa las paradas en cada ruta	Asigna las paradas a cada ruta	Asigna y programa las paradas para cada ruta	Asigna las paradas a los almacenes	Asigna y programa las paradas según distancias en callejero	Incorpora información en tiempo real sobre tráfico	Considera diferentes velocidades según el momento del día	Incorpora modelos de simulación probabilística para cargas y tiempos	Ventanas de tiempo rígidas	Ventanas de tiempo flexibles
ArcLogistics Route 3	sí	sí	sí	sí	sí	-	-	sí	sí	sí
Compass	sí	sí	sí	sí	-	-	-	sí	sí	sí
Direct Route	sí	sí	sí	sí	sí	-	-	sí	sí	-
Edgar Transportation Management System	sí	sí	sí	sí	sí	sí	sí	sí	sí	-
Fleetwise	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
ILOG Dispatcher	sí	sí	sí	sí	sí	-	-	-	sí	sí
Intertour/Interload	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
Manugistics Fleet Management	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
Optrak4	sí	sí	sí	sí	sí	sí	-	sí	sí	-
Prophesy Mileage & Routing	sí	-	-	-	-	-	-	-	-	-
Protour	sí	sí	sí	-	sí	-	sí	sí	sí	sí
RoadNet 5000	sí	sí	sí	-	sí	-	sí	sí	sí	-
Roadshow System	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
RoutePro	sí	sí	sí	sí	sí	-	sí	sí	sí	sí
RouteSmart	sí	sí	sí	sí	sí	-	sí	sí	sí	-
Routronic 2000	sí	sí	sí	sí	-	-	-	-	sí	sí
SHORTREC product suite	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
STARS	sí	sí	sí	sí	sí	-	sí	-	sí	sí
Territory Planner	sí	sí	sí	-	sí	-	sí	sí	sí	-
tmsRouter	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
tmsZoneDesigner	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
Trapeze	sí	sí	-	-	sí	sí	sí	sí	sí	sí
TruckSTOPS	sí	sí	sí	sí	sí	-	-	sí	sí	-
VersaTrans	sí	Sí	sí	sí	sí	-	-	sí	sí	sí

Proveedor	Integración con sistemas GIS			Compatibilidad con los siguientes proveedores de mapas				Complementos al software					
	Muestra rutas y paradas en un mapa	Edita las rutas a través de "arrastre"	Geocidifica en base a direcciones	Etak/TeleAtlas	GDT	Nav Tech	Tiger	Dispositivo electrónico para vehículo	Servicio mensajería inalámbrica	Seguimiento de vehículos en tiempo real	Scanner código de barras	Software de gestión de la cadena de valor (P.e. gestión inventarios)	Proceso pedidos de clientes
ESRI	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
RouteLogic	sí	-	sí	-	sí	-	sí	sí	sí	sí	sí	-	sí
Appian Logistics Software, Inc.	sí	sí	sí	sí	sí	sí	sí	-	-	-	-	-	-
Edgar Management Consultants, Inc.	sí	-	sí	-	-	-	sí	sí	sí	sí	sí	sí	sí
Descartes Systems Group	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	-	-
ILOG	sí	-	-	-	-	-	-	-	-	-	-	sí	-
Intertour/Interload	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí
Manugistics, Inc.	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	-	sí	sí
Optrak Distribution Software	sí	sí	sí	-	-	sí	-	sí	sí	sí	sí	-	-
Prophesy Transportation Solutions, Inc.	sí	sí	sí	-	-	-	-	-	-	-	-	-	-
Prologos Planung und Beratung	sí	sí	sí	sí	-	sí	-	-	-	-	-	sí	-
UPS Logistics Group	sí	sí	sí	-	sí	sí	-	sí	sí	sí	sí	-	-
Descartes Systems Group	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	-	-
CAPS Logistics	sí	sí	sí	sí	sí	sí	-	sí	sí	sí	sí	sí	sí
RouteSmart Technologies, Inc.	sí	sí	sí	sí	sí	sí	sí	-	-	-	-	-	-
Carrier Logistics, Inc.	sí	-	sí	-	-	-	sí	sí	sí	sí	sí	sí	sí
ORTEC	sí	sí	sí	sí	n	sí	n	sí	sí	sí	n	sí	sí
SAITECH, Inc.	sí	sí	sí	-	-	-	sí	-	-	sí	-	sí	sí
UPS Logistics Group	sí	sí	sí	-	sí	sí	-	sí	sí	sí	sí	-	-
GeoComtms	sí	sí	sí	-	sí	sí	-	sí	sí	sí	-	-	-
GeoComtms	sí	sí	sí	-	sí	sí	-	sí	sí	sí	-	-	-
Trapeze Software Group, Inc.	sí	sí	sí	sí	sí	sí	sí	sí	sí	sí	-	-	-
Micro Analytics, Inc.	sí	sí	sí	sí	sí	-	sí	-	sí	-	-	sí	-
VersaTrans Solutions, Inc.	sí	sí	sí	-	sí	sí	sí	-	-	-	-	-	-

Anexo II

Código del Programa Comentado

En este anexo se presentan las secuencias de código de programa relativas a los procedimientos de cálculo recogidos en el algoritmo, así como la presentación de la interfaz de usuario, con la finalidad de fomentar y facilitar posteriores desarrollos del código en la comunidad científica.

1 Código del algoritmo comentado

En este apartado se describe la fase de programación del algoritmo, destacando la construcción de los procedimientos más importantes del código. Antes de proceder a las explicaciones, es necesario comentar algunas cuestiones relevantes sobre el programa desarrollado, y que se han de tener en cuenta en la fase posterior de experimentación y análisis de resultados.

En primer lugar se trata de un programa *stand-alone*, por lo que se han tenido que realizar tareas adicionales encaminadas al diseño del entorno operativo del software, pasarelas de datos entre la interfaz y el módulo de computación y viceversa, así como de todas las acciones requeridas por el usuario como *abrir archivos*, *borrar*, *guardar*, etc. En este sentido el software presentado en este trabajo tiene la ventaja de que no requiere de ninguna aplicación soporte por lo que es fácilmente transferible para su evaluación y testeo, en aras de profundizar sobre la investigación aquí realizada.

En cuanto a la elección del lenguaje de programación escogido se ha optado por la elección de *Delphi 5*, una versión avanzada de *Pascal*, que permite una flexibilidad y adapatibilidad absoluta a las necesidades del problema, además de su facilidad de uso, y su velocidad de compilación.

Delphi se presenta como un entorno de desarrollo rápido de aplicaciones (*RAD*), con un potente lenguaje el *Object Pascal*, un compilador muy rápido que permite crear ejecutables con una velocidad cercana al *C++,* y con múltiples posibilidades: bases de datos, multimedia, web, etc. Desciende del lenguaje desarrollado por Niklaus Wirth en 1971, para facilitar la enseñanza de la programación estructurada, y que le puso el nombre del filósofo y matemático francés Pascal. En aquella época, las fases de edición de código, compilación y enlazado, iban por separado, hasta que Philippe Khan, el fundador de *Borland*, actualmente *Inprise*, creo el *Turbo Pascal*, el primer entorno de programación DOS, con el editor, compilador y linkador integrados, siguiéndole los famosos *Turbo Basic* y *Turbo C/C++.* Posteriormente vieron la luz las versiones de *Turbo Pascal* para *Windows*, pero en 1995 ante la aparición de los sistemas *Windows* a 32 bits, *Borland* decide dar un giro al producto, lo potencia y mejora incorporando las nuevas tecnologías Windows, lo denomina *Delphi*.

En las próximas líneas se profundiza en el desarrollo del código para la ejecución del algoritmo presentado en el apartado anterior.

1.1 Definición de variables

En esta sección se definen las variables a ser utilizadas por el código, y que se corresponden con las variables enunciadas en el modelo matemático. Igualmente se establece el tipo de variable al que hace referencia cada una de las mismas.

TRuta	Tipo que define una ruta
Nombre	Nombre de la ruta
Tiempo	Tiempo total
Distancia	Distancia total
Carga	Carga que ha entregado
Espera	Espera total
Nodo	Lista de los nodos de la ruta
TnodoInsSenc	Tipo para los nodos susceptibles de una inserción sencilla
Nodo	Indicativo de un nodo en consideración
Holgura	Valor de una holgura o tiempo de espera
TNodoInsDoble	Tipo para los nodos susceptibles de una inserción doble
NodoIni	Indicativo de un nodo inicial en una secuencia
NodoFin	Indicativo del nodo final en una secuencia
Holgura	Valor de una holgura o tiempo de espera
num_nodos	Número de nodos del problema, sin contar el depósito
num_camiones	Número de camiones del problema
carga_cam	Carga máxima para cada camión
num_rutas	Número de rutas existentes, que aumentará según las necesidades
num_rutas_ini	**N**úmero de rutas con las que empieza el problema
beta	Parámetro que define la desviación máxima de cada camión respecto a su ruta
gamma	Parámetro ajustable por el usuario, del tipo de β
distancia_total	Suma de las distancias de todas las rutas
lista_asig	Lista con los nodos asignados : contiene 'true' si está asignado y 'false' si está libre, siendo el identificador dentro de la lista el mismo que el del nodo
lista_rutas	Lista de las rutas
lista_criticos	Lista de los nodos a insertar
lista_ins_senc	Lista de los posibles nodos para realizar una inserción sencilla antes de los de la lista crítica
lista_ins_doble	Lista de los posibles nodos para realizar una inserción doble antes de los de la lista crítica
matriz_nodos	Matriz que almacena los datos de cada nodo (ver Set_Nodo y Set_Fila)
matriz_dist	Matriz con las distancias entre nodos

Figura 1. Lista de variables a utilizar en el algoritmo

1.2 Procedimientos de cálculo incluidos en el algoritmo

El algoritmo presentado en este trabajo contempla dos posibilidades de cálculo de la solución final. En primer lugar se establece la posibilidad de cálculo de una solución individual a través del ajuste por parte del usuario de los tres parámetros básicos del problema. Este procedimiento es sumamente rápido, a pesar de la gran cantidad de cálculos que ha de realizar el algoritmo. De esta manera se consigue obtener una solución determinista a los datos del problema, reportando la mejor solución posible con todos los datos preestablecidos.

En segundo lugar, y teniendo en cuenta que las soluciones individuales varían en función de los parámetros prefijados, se ha establecido un segundo procedimiento para validar la bondad de ajuste de estos parámetros por parte del usuario. En este caso se trata de la búsqueda de la mejor solución a través de la exploración de todas las combinaciones posibles de los tres parámetros incluidos en el algoritmo. Estos parámetros recordamos que son:

- **Número inicial de rutas o nodos semilla en consideración:** Puede variar entre 1 y 25
- **Parámetro β:** Puede variar entre 1 e infinito, aunque en este caso se ha limitado a 5.
- **Parámetro γ:** Puede variar entre 1 e infinito, y al igual que en el caso anterior se ha limitado a 5.

En este caso el algoritmo explora todas las soluciones posibles para cualquier combinación de estos tres parámetros, donde el número de nodos semilla oscila de forma discreta por enteros, y los parámetros β y γ son continuos, aunque en este caso se hacen oscilar de forma discreta a través de incrementos centesimales. En este sentido el número total de exploraciones que realiza el algoritmo es de *25 x 400 x 400*, es decir un total de *4.000.000* exploraciones. De este número total de exploraciones, el algoritmo reporta la mejor solución encontrada, así como los valores indicativos de los parámetros para los que se encontró dicha solución.

1.2.1 Cálculo de una solución individual

A continuación se establece el orden de instrucciones a seguir para obtener una solución individual del problema.

```
procedure Iniciar_Problema ; stdcall;
var
   i : integer;
   distancia : real;
begin
     Set_Distancias;                                                    (1)
     Inic_Lista_Asig;                                                   (2)
     Inic_Lista_Rutas;                                                  (3)
     Generar_Lista_Criticos_Primera;                                    (4)
     Asignar_Nodos_Primera;                                             (5)
     while Hay_Nodos_Libres do
     begin
          Generar_Lista_Criticos;                                       (6)
          Asignar_Nodos;                                                (7)
     end;
     Finalizar_Rutas;                                                   (8)
     distancia := 0;
     for i:=1 to num_rutas do
          distancia := distancia + lista_rutas[i-1].Distancia;
     distancia_total := distancia;                                      (9)
end;
```

Figura 2. Procedimiento de cálculo de solución individual

Una vez que el usuario selecciona el botón de "*Calcular*", se ejecuta el cálculo de la solución tal y como se describe aquí. Para ello en primer lugar se calculan las distancias para cada par de nodos contemplados en el problema, según las coordenadas de los mismos (1). A continuación se inicializan las listas de nodos, en la que se incluyen todos los nodos del problema acompañados de una variable booleana que indican si están o no asignados, y la lista de rutas, donde se representan los nodos incluidos en cada ruta, así como todos los datos relevantes para cada una de las mismas. Seguidamente el algoritmo selecciona la lista de los primeros nodos a ser asignados, de forma que asigna cada uno a una ruta diferente según (4) y (5). En la asignación a las rutas existentes, se procede a

contemplar si es posible realizar alguna inserción intermedia, ya sea de un solo nodo o de dos nodos simultáneamente antes de realizar la visita de ese nodo crítico (Estos procedimientos se describen con detalle en los siguientes apartados). Después de realizar la primera asignación, y mientras queden nodos no asignados se seleccionan nuevas listas de nodos críticos para realizar sus adiciones a las mejores rutas en consideración, tal y como se contempla en (6) y (7). Por último, cuando ya no queden nodos por asignar, entonces se hace regresar a todos los vehículos al depósito central dentro del plazo previsto, como indica (8). Por último el algoritmo computa la distancia total recorrida por todas las rutas, como la suma de las distancias recorridas por cada uno de los vehículos en consideración (9).

Este procedimiento aquí descrito tiene su desarrollo en otros tantos procedimientos a los que hace referencia en su estructura, y que son explicados con todo detalle en las próximas líneas.

1.2.2 Cálculo de la mejor solución

En este caso se contempla el procedimiento relativo a la búsqueda de la mejor solución posible por parte del algoritmo. El procedimiento que sigue también es altamente intuitivo, y se resume en la búsqueda de la mejor combinación de los tres parámetros de usuario para un problema concreto. A continuación se presentan las instrucciones correspondientes al control de "*Calcular Óptimo*" incluido en la interfaz.

En esta secuencia de instrucciones se contempla el contador triple mencionado anteriormente y que se recoge en las expresiones con la notación (10). De esta forma se hacen variar los tres parámetros para considerar todas las combinaciones posibles. Para cada una de estas combinaciones se establece el cálculo de la solución individual, tal y como se recoge en el apartado anterior, y que se desarrolla entre las notaciones (11) y (12). Finalmente se computa la distancia total para la solución en cuestión, y en el caso de que sea mejor que la solución anterior, entonces se almacena en memoria para realizar la siguiente comprobación (14). Este procedimiento se termina cuando se finalizan todos los contadores, de manera que la solución devuelta es la mejor en cuanto a la distancia recorrida.

```
procedure Mejor_Solucion ; stdcall;
var
```

```
   rutas_ini, mejor_num_rutas, i : integer;
   mejor_beta, mejor_gamma, mejor_distancia, distancia : real;
begin
     Set_Distancias;
     mejor_distancia := matriz_dist[0,0];
     mejor_num_rutas := num_rutas;
     mejor_beta := beta;
     mejor_gamma := gamma;
     for rutas_ini:=1 to 25 do                                              (10)
     begin
          num_rutas := rutas_ini;
          beta := 1;
          while (beta <= 5) do                                              (10)
          begin
               gamma := 1;
               while (gamma <= 5) do
                                                                            (10)
               begin
                    num_rutas := rutas_ini;                                 (11)
                    Inic_Lista_Asig;
                    Inic_Lista_Rutas;
                    Generar_Lista_Criticos_Primera;
                    Asignar_Nodos_Primera;
                    while Hay_Nodos_Libres do
                    begin
                         Generar_Lista_Criticos;
                         Asignar_Nodos;
                    end;
                    Finalizar_Rutas;
                    distancia := 0;
                    for i:=1 to num_rutas do
                        distancia := distancia + lista_rutas[i-1].Distancia;  (12)
                    if   (distancia   <   mejor_distancia)   and   (num_rutas   <=
num_camiones) then                                                          (13)
                    begin
                         mejor_distancia := distancia;
                         mejor_num_rutas := rutas_ini;
                         mejor_beta := beta;
                         mejor_gamma := gamma;
                    end;
                    gamma := gamma + 0.01;                                  (10)
               end;
               beta := beta + 0.01;                                         (10)
          end;
     end;
     num_rutas := mejor_num_rutas;
     num_rutas_ini := num_rutas;
     beta := mejor_beta;
     gamma := mejor_gamma;
end;
```

Figura 3. Procedimiento de cálculo de la mejor solución

1.3 Reglas y procedimientos del algoritmo

En este apartado se establece el conjunto de procedimientos que integran el código del algoritmo y que se presenta como el motor de cálculo de todo el software. Para ello se presentan aquí los enunciados de cada uno de los algoritmos que posteriormente se describen en los siguientes apartados, y que configuran las instrucciones de cálculo contenidas en los dos procedimientos principales descritos anteriormente.

1.3.1 Procedimiento de cálculo de la matriz de distancias

El primer paso del algoritmo establece el procedimiento de cálculo de distancias tal y como se define en el apartado anterior. Para ello se recogen los datos de la matriz de datos originales ya importada por la aplicación, o bien volcada por el usuario en caso de tratarse de problemas específicos.

```
procedure Set_Distancias;
var
   i, j : integer;
begin
     SetLength(matriz_dist,num_nodos+1,num_nodos+1);
     for i:= 0 to num_nodos do
         matriz_dist[i,i]:= MaxInt;                                          (14)
     for i:= 1 to num_nodos do
         for j:= 0 to i - 1 do
             begin
             matriz_dist[i,j]:=                          sqrt(sqr(matriz_nodos[i,1]-
matriz_nodos[j,1])+sqr(matriz_nodos[i,2]-matriz_nodos[j,2]));               (15)
             matriz_dist[j,i]:= matriz_dist[i,j];                            (16)
             end;
end;
```

Figura 4. Procedimiento de cálculo de la matriz de distancias

Se trata de una matriz simétrica donde la distancia entre el nodo *i* y *j* es igual a la distancia entre el nodo *j* e *i* (16). Además se ha provocado artificialmente que la diagonal principal sea un valor muy elevado (14). Esto obedece a que la diagonal principal es siempre *0*, por lo que en las elecciones de las menores distancias siempre se unirían los nodos consigo mismo. Para evitar que esto ocurra, se provoca un rechazo artificialmente otorgando un valor lo suficientemente alto (*MaxInt*), equivalente a infinito. El valor de las distancias entre cada dos nodos se obtiene a través de la distancia euclidea formulada por Pitágoras (15).

1.3.2 Creación de la lista de nodos *L0*

Una vez que se obtienen las distancias o tiempos de desplazamiento entre todos y cada uno de los nodos incluido el depósito central, entonces se procede a la consideración de los nodos que integran *L0*. Estos nodos son aquellos que están pendientes de asignación, y que al comienzo de la planificación son todos los nodos del problema original, aunque si bien se han integrado todos los nodos en todo momento y se ha establecido una variable booleana para saber si están o no asignados, en vez de retirarlos de la lista. De esta manera la variable tomará el valor "*verdadero*" si el nodo en cuestión ya ha sido asignado, y "*falso*" si está pendiente de asignación. La creación de la lista *L0* se corresponde con el siguiente procedimiento. Se provoca que todos los nodos estén pendientes de asignación, salvo el nodo *0*, que se asignará a todas las rutas.

```
procedure Inic_Lista_Asig;
var
   i : integer;
begin
     SetLength(lista_asig,num_nodos+1);
     lista_asig[0] := true;
```

```
     for i:= 1 to num_nodos do
         lista_asig[i] := false;
end;
```

Figura 5. Generación de la lista de nodos *L0*

1.3.3 Creación de la lista de rutas y sus variables

A continuación se genera el conjunto de rutas a ser considerado y se procede a la inicialización de las R rutas de partida a ser consideradas inicialmente por el problema según las definiciones que ha realizado el usuario en la interfaz. Para ello se establecen todas las variables de cada una de las rutas en consideración, tal y como se definían anteriormente, y se les concede un valor inicial de *0*.

```
procedure Inic_Lista_Rutas;
var
   i : integer;
begin
     for i:= 1 to num_rutas do
         Inic_Ruta(i);
end;
```

Figura 6. Procedimiento de inicialización de la lista de rutas

```
procedure Inic_Ruta(i : integer);
begin
     SetLength(lista_rutas,i);
     lista_rutas[i-1].Nombre := 'R' + inttostr(i);
     lista_rutas[i-1].Tiempo := 0;
     lista_rutas[i-1].Distancia := 0;
     lista_rutas[i-1].Carga := 0;
     lista_rutas[i-1].Espera := 0;
     SetLength(lista_rutas[i-1].Nodo,1);
     lista_rutas[i-1].Nodo[0] := 0;
end;
```

Figura 7. Procedimiento para inicializar los valores de una ruta

1.3.4 Generación de la primera lista de nodos semilla *L1*

Una vez dispuestos todos los datos necesarios para abordar la ejecución del algoritmo, así como preparadas e inicializadas las primeras *R* rutas que se van a considerar, entonces se procede con las instrucciones de decisión del propio algoritmo. En primer lugar se genera la lista *L1* con los *R* primeros nodos críticos a ser estudiados. Esta lista procede de *L0*, y una vez seleccionada se marcarán los nodos como ya asignados.

El procedimiento trabaja de la siguiente manera, en primer lugar calcula la holgura para todos los nodos del problema y que no están asignados incluidos en la lista *L0*, identificando el menor. Esto se realiza a través del siguiente procedimiento.

```
function Buscar_Menor_Primera(menor : real) : integer;
var
   aux_nodo, i : integer;
   aux_t : real;
begin
     aux_t := matriz_dist[0,0];     aux_nodo := 0;
     for i:= 1 to num_nodos do
```

```
         if not lista_asig[i] then
            if     ((matriz_nodos[i,5]-matriz_dist[0,i])     >=     menor)     and
((matriz_nodos[i,5]-matriz_dist[0,i]) < aux_t) then
               begin
                 aux_nodo := i;
                 aux_t := matriz_nodos[i,5]-matriz_dist[0,i];
               end;
     result := aux_nodo;
end;
```

Figura 8. Cálculo de la menor holgura

En esta función se calcula el nodo cuya holgura es menor para ser integrado en la lista *L1* inicial de nodos críticos. Este calculo se realiza a través de la comparación del tiempo de cierre de cada nodo con el tiempo necesario para llegar al mismo desde el depósito. A continuación, este nodo se pasa a la lista *L1*, y se vuelve a buscar el siguiente nodo hasta completar el total de la lista *L1*. De esta manera se van eliminando los nodos que pasan a forma parte de *L1* de la lista original *L0*, cambiando el valor de la variable booleana a "*verdadero*", indicando que el nodo ya ha sido asignado.

```
procedure Generar_Lista_Criticos_Primera;
var
   i : integer;
begin
     SetLength(lista_criticos,num_rutas);
     lista_criticos[0] := Buscar_Menor_Primera(0);
     lista_asig[lista_criticos[0]] := true;
     for i:= 2 to num_rutas do
       begin
         lista_criticos[i-1]                                                    :=
Buscar_Menor_Primera(matriz_nodos[lista_criticos[i-2],5]-
matriz_dist[0,lista_criticos[i-2]]);
         lista_asig[lista_criticos[i-1]] := true;
       end;
end;
```

Figura 9. Creación de la lista *L1* de nodos críticos

1.3.5 Procedimiento de asignación de nodos de la primera lista *L1*

En el caso de la generación de la primera lista de nodos críticos del problema se procede a la asignación directa de cada uno de estos nodos a una ruta diferente de las previamente inicializadas. En principio solamente se realiza la asignación de cada nodo a una ruta diferente. Posteriormente se estudia particularmente la adición de cada uno de los nodos a sus rutas correspondientes para establecer si la adición se realiza directamente, con una escala intermedia o con dos escalas intermedias.

```
procedure Asignar_Nodos_Primera;
var
   i : integer;
begin
     for i:= 0 to num_rutas-1 do
         Definir_Ruta(i+1,lista_criticos[i]);
     SetLength(lista_criticos,0);
```

```
end;
```

Figura 10. Asignación de los primeros nodos semilla

1.3.6 Cálculo de la segunda y siguientes listas de nodos críticos *L1*

En estos momentos se han seleccionado los *R* nodos críticos que forman parte de la lista inicial *L1* en la primera etapa del algoritmo. Sin embargo, si en vez de la primera fase, se tratase de la segunda o posteriores fases, entonces el cálculo sería diferente, dado que en vez de las holguras resultantes, se tendrían en cuenta los tiempos de cierre de cada nodo para configurar la lista *L1*.

```
function Buscar_Menor(menor : integer) : integer;
var
   aux_nodo, aux_t, i : integer;
begin
     aux_t := matriz_nodos[0,5];
     aux_nodo := 0;
     for i:= 1 to num_nodos do
         if not lista_asig[i] then
            if (matriz_nodos[i,5] >= menor) and (matriz_nodos[i,5] < aux_t) then
               begin
                 aux_nodo := i;
                 aux_t := matriz_nodos[i,5];
               end;
     result := aux_nodo;
end;
```

Figura 11. Búsqueda del nodo con momento de cierre más temprano

A continuación, una vez detectado este nodo crítico se pasa a la lista *L1*, y se vuelve a buscar el siguiente menor, hasta completar un total de *R* nodos, donde *R* ya no es el número inicial de rutas, sino que representa el total de rutas que actualmente se están planificando. Al igual que en el caso anterior, se cambia la variable booleana para indicar que son nodos ya asignados en el problema.

```
procedure Generar_Lista_Criticos;
var
   i : integer;
begin
     SetLength(lista_criticos,num_rutas);
     lista_criticos[0] := Buscar_Menor(0);
     lista_asig[lista_criticos[0]] := true;
     for i:= 2 to num_rutas do
       begin
         lista_criticos[i-1] := Buscar_Menor(matriz_nodos[lista_criticos[i-2],5]);
         lista_asig[lista_criticos[i-1]] := true;
       end;
end;
```

Figura 12. Procedimiento de generación de la segunda y posteriores listas *L1*

1.3.7 Procedimiento de asignación de los nodos de *L1* a las rutas

En este momento se procede al estudio de los nodos incluidos en la lista *L1* de nodos críticos a estudiar. Para ello se aplica el procedimiento de asignación a rutas de cada uno de los nodos en consideración. Este procedimiento se desarrolla de la siguiente manera:

```
procedure Asignar_Nodos;
var
   i, j, fin, cab_ruta, ruta, long : integer;
   dist, holgura : real;
begin
     fin := num_rutas - 1;
     dist := matriz_dist[0,0];
     for i:=0 to fin do                                                    (17)
         if lista_criticos[i] <> 0 then
         begin
              ruta := 0;
              for j:=0 to num_rutas-1 do                                   (18)
              begin
                   long := length(lista_rutas[j].Nodo);
                   cab_ruta := lista_rutas[j].Nodo[long-1];                (19)
                   holgura   :=   matriz_nodos[i,5]   -   lista_rutas[j].Tiempo   -
matriz_dist[cab_ruta,lista_criticos[i]];                                   (20)
                   if       (holgura      >=      0)       and       (dist       >
matriz_dist[cab_ruta,lista_criticos[i]]) then                              (21)
                        if Comprobar_Nodo(lista_criticos[i],j+1,cab_ruta) then (22)
                        begin
                             dist := matriz_dist[cab_ruta,lista_criticos[i]];  (21)
                             ruta := j + 1;
                        end;
              end;
              dist := matriz_dist[0,0];
              if ruta = 0 then
                 begin
                      num_rutas := num_rutas + 1;                          (23)
                      Inic_Ruta(num_rutas);
                      Definir_Ruta(num_rutas,lista_criticos[i]);           (24)
                 end
              else
                  Definir_Ruta(ruta,lista_criticos[i]);                    (24)
         end;
end
```

Figura 13. Procedimiento de asignación de los nodos de *L1* a las rutas

En este procedimiento se establece la asignación de los nodos incluidos en la lista de nodos críticos *L1* a las rutas existentes. Para ello se procede de la siguiente manera, en primer lugar (17), para cada uno de los nodos críticos incluidos en *L1* se realiza el cálculo de los tiempos estimados de llegada desde cada una de las rutas en consideración (18). Estos tiempos de llegada son la distancia entre cada una de las cabezas de ruta (19) y cada uno de los nodos en consideración. Con estos tiempos estimados de llegada se procede al cálculo de las holguras resultantes ante la visita de cada uno de los vehículos que están sirviendo las rutas, y cada uno de los nodos críticos en consideración. Estas holguras se calculan en la expresión (20). Formalmente para una ruta *j* y un nodo *i* se establece el siguiente cálculo: $tr_j + t_{CRj\,i} \leq tc_i$

Es decir que la consideración de visita de la ruta *j* al nodo *i,* ha de permitir la llegada antes del momento de cierre al nodo *i* en consideración. Incluyendo la correspondiente variable de holgura, la ecuación resulta en (20), de manera que solamente serán posibles las visitas con holguras mayores o iguales a *0*, para cualquier ruta *j* y nodo crítico *i.*

En este sentido, para un determinado nodo crítico *i*, se revisa su posible adición a todas y cada una de las rutas en consideración, de manera que solamente se podrá añadir a aquellas para las que la holgura es positiva. Además interesa saber de entre todas esas rutas, cuál es la mejor opción. En este caso el algoritmo se ha diseñado para que de entre todas las posibles, solamente se tenga en cuenta aquella ruta de cuya cabeza está más cerca, es decir la ruta más cercana, dentro de todas las posibles. Por ello se computa para todas las opciones con holguras positivas (21), aquella que tiene una menor distancia entre el nodo crítico a ser añadido y la cabeza de esa ruta (21). El algoritmo almacena los datos del primer nodo y de la primera ruta en cuestión, y a partir de ahí procede a revisar que ocurre con las siguientes rutas, tal y como se recoge en (22). En este punto se llama a una función para que verifique las opciones de otras rutas. Para ello se comprueba para las siguientes rutas que se cumplen todas las restricciones:

- **Caso 1:** Si no existen vehículos a los que se pueda asignar el nodo porque todos llegarían tarde, lo cual equivale a una holgura negativa, entonces se procede a generar una nueva ruta según (23).
- **Caso 2**: Igualmente si la asignación implica una sobrecarga del camión, entonces también se genera una nueva ruta según (23)
- **Caso 3:** También si la asignación provocase un retraso en el regreso al depósito, entonces también se generaría una nueva ruta según (23)

Sin embargo si todas las restricciones se cumplen el algoritmo sustituye los valores de la anterior ruta, por los valores de la nueva ruta, y procede de esta manera hasta que revisa todas y cada una de las rutas, almacenando la mejor opción, que posteriormente dará lugar a la asignación del nodo *i* a la ruta finalmente escogida. Este procedimiento, asigna el nodo *i* a la ruta *j* de acuerdo con (24) que se encuentre más cercana, a no ser que se viole alguna de las restricciones establecidas en el procedimiento de la figura 14. Para las asignaciones de los siguientes nodos de *L1* se tiene en cuenta la actualización de las rutas que previamente han sido ampliadas.
En la generación de una nueva ruta según (23) se procede creando una nueva ruta que será tenida en cuenta en sucesivas pasadas para el resto de los nodos de *L1* pendientes de asignación.

Las restricciones contempladas en los tres casos anteriores derivan de los valores de la siguiente variable booleana:

```
function Comprobar_Nodo(nodo, ruta, cab_ruta : integer) : boolean;
var
   aux : boolean;
begin
     aux := true;
     if (lista_rutas[ruta-1].Tiempo + matriz_dist[cab_ruta,nodo]
```

```
        +  matriz_nodos[nodo,6]  +  matriz_dist[nodo,0])  >  matriz_nodos[0,5]  then
                                                                                (25)
        aux := false;
     if     (lista_rutas[ruta-1].Tiempo     +     matriz_dist[cab_ruta,nodo])     >
matriz_nodos[nodo,5] then                                                       (26)
        aux := false;
     if (lista_rutas[ruta-1].Carga + matriz_nodos[nodo,3]) > carga_cam then   (27)
        aux := false;
     result := aux;
end;
```

Figura 14. Función de comprobación de cumplimiento con restricciones

Esta función tiene la finalidad de examinar todas y cada una de las rutas, para un nodo concreto *i*, de manera que cada vez que se selecciona un nodo *i* de la lista de críticos, entonces el algoritmo procede a revisar los valores de la holgura resultante para cada una de las rutas en cuestión, manteniendo siempre la mejor solución (i.e. la que implica menor distancia entre cabeza de ruta y nodo a ser asignado) en la memoria, pero contrastando las posibilidades que ofrecen todas las rutas a través de la comprobación de si se violan o no las restricciones impuestas en la asignación y que se recogen a continuación:

Si la asignación de ese nodo a la ruta en cuestión, implica un tiempo de llegada al depósito superior al momento de cierre del mismo, entonces no se realizará la asignación a esa ruta. Esta restricción se representa en (25), y formalmente implica que si se cumple que $tr_j + t_{CRj\,i} + ts_i + t_{i0} > tc_0$ .Entonces la variable booleana devuelve un valor "*falso*", que corta la ejecución del bucle en cuestión, seleccionando la ruta anterior.

La segunda excepción hace referencia al cálculo de la holgura para la nueva ruta. En este caso se mide si desde la cabeza de la nueva ruta da tiempo a servir el nodo *i* o no. Por ello se establece que $tr_j + t_{CRj\,i} \leq tc_i$. Si se viola esta restricción entonces se corta el bucle y nos quedamos con la ruta anterior

La tercera excepción deriva de la no posibilidad de violación de la restricción de las cargas de los camiones. En este sentido en (27) se contempla que si al asignar ese nodo a la ruta se viola la capacidad total del vehículo, entonces la variable booleana toma un valor de "*falso*", cortando el bucle en cuestión, y escogiendo el algoritmo la ruta anterior.

El resultado de esta función, es el valor de la variable booleana, "*verdadero*" si se respetan todas las restricciones, o "*falso*" si incumple alguna de ellas, cortando el bucle.

1.3.8 Adición de un nodo a una ruta

En este procedimiento se detallan las reglas de adición e inserción. Quizás sea esta la parte más compleja del algoritmo, dado que desde esta estructura es desde donde se llama a otros procedimientos para realizar todos los cálculos necesarios. En el apartado anterior se establecían las instrucciones relativas a la asignación de un nodo a una ruta de terminada. En este procedimiento que sigue se establece como ha de llevarse a cabo la adición de ese nodo a la ruta, directamente, con una inserción sencilla, o doble.

```
procedure Definir_Ruta(ruta, nodo_crit : integer);
var
   menor, cab_ruta, long : integer;
   holgura : real;
begin
     menor := Hallar_Minimo_T_Serv;                                          (28)
     long := length(lista_rutas[ruta-1].Nodo);
     cab_ruta := lista_rutas[ruta-1].Nodo[long-1];
     holgura := matriz_nodos[nodo_crit,5] - matriz_dist[cab_ruta,nodo_crit]; (29)
     if holgura < menor then                                                 (30)
        Asignar_Nodo_Ruta(nodo_crit,ruta)                                    (31)
     Else
         begin
              Generar_Lista_Ins_Senc(nodo_crit,ruta);                        (32)
              if (length(lista_ins_senc) = 0) then                           (33)
                 Asignar_Nodo_Ruta(nodo_crit,ruta)
              else
                  if Comprobar_Holguras_L2 then                              (34)
                     begin
                          Generar_Lista_Ins_Doble(nodo_crit,ruta);           (35)
                          if (length(lista_ins_doble) > 0) then
                              begin

Asignar_Nodo_Ruta(lista_ins_doble[0].NodoIni,ruta);                          (36)
                                  lista_asig[lista_ins_doble[0].NodoIni] := true;
Asignar_Nodo_Ruta(lista_ins_doble[0].NodoFin,ruta);
                                  lista_asig[lista_ins_doble[0].NodoFin] := true;
                              end
                           else
                              begin
Asignar_Nodo_Ruta(lista_ins_senc[0].Nodo,ruta);     (37)
                                   lista_asig[lista_ins_senc[0].Nodo] := true;
                              end;
                           Asignar_Nodo_Ruta(nodo_crit,ruta);
                     end
                  else
                      begin
                           Asignar_Nodo_Ruta(lista_ins_senc[0].Nodo,ruta);    (38)
                           lista_asig[lista_ins_senc[0].Nodo] := true;
                           Asignar_Nodo_Ruta(nodo_crit,ruta);
                      end;
         end;
end;
```

Figura 15. Procedimiento general de adición de un nodo de L1 a una ruta preestablecida

A continuación se detallan todos los procedimientos a los que hace referencia esta regla general, así como también se desarrollan las funciones a las que hace referencia y que son necesarias para la ejecución del mismo.

1.3.8.1 Cálculo del menor tiempo de servicio (Instrucción 28)

En primer lugar es necesario computar cuál es el menor valor del los tiempos de servicio considerados para todos los nodos, dado que se va a utilizar en las próximas instrucciones como un valor de referencia.

La siguiente función devuelve un valor que es necesario para realizar los cálculos del algoritmo. Si bien este valor es conocido de antemano, dadas las características del problema en cuestión, podría ser que el algoritmo trabajase con tiempos de servicio

diferentes, lo que complicaría enormemente los cálculos, y ralentizaría el cómputo global del programa. Por ello aún siendo un valor conocido a priori, se ha establecido la siguiente función para calcular el tiempo de servicio mínimo de entre todos los nodos del problema.

```
function Hallar_Minimo_T_Serv : integer;
var
   i, aux : integer;
begin
     aux := matriz_nodos[1,6];
     for i:= 2 to num_nodos do
         if matriz_nodos[i,6] < aux then
            aux := matriz_nodos[i,6];
     result := aux;
end;
```

Figura 16. Función de cálculo del menor tiempo de servicio

A partir de ahí se procede al cálculo de la holgura existente en esa asignación (29). En este momento el algoritmo calcula la holgura del nodo en estudio, tal y como veíamos al comienzo. Para ello se tiene en cuenta el momento de cierre del nodo en estudio y el tiempo necesario de llegada desde la ruta a la que ha sido asignado y este nodo. Una vez calculada la holgura, entonces el algoritmo procede de diferentes maneras. En primer lugar, si la holgura es menor que el menor de los tiempos de servicio, entonces se procede a la adición directa del nodo a la cabeza de ruta correspondiente. Sin embargo si la holgura es mayor, entonces existe la posibilidad de realizar alguna parada intermedia para visitar uno o dos de los nodos que se encuentren en el camino.

1.3.8.2 Adición directa de un nodo a una ruta (Instrucción 31)

Si la holgura resultante es menor que el mínimo tiempo de servicio, entonces no es posible realizar ningún tipo de visitas intermedias, por lo que se procede a añadir directamente el nodo en cuestión a la ruta previamente seleccionada tal y como establecen las instrucciones (30) y (31). En este procedimiento se establece la asignación del nodo a la ruta, con el recálculo de todas las variables de la misma: Tiempo de la ruta, Distancia de la ruta, Carga de la ruta, Espera de la ruta.

```
procedure Asignar_Nodo_Ruta(nodo, ruta : integer);
var
   cab_ruta, ultimo : integer;
begin
     ultimo := length(lista_rutas[ruta-1].Nodo);
     cab_ruta := lista_rutas[ruta-1].Nodo[ultimo-1];
     lista_rutas[ruta-1].Tiempo       :=       lista_rutas[ruta-1].Tiempo       +
matriz_dist[cab_ruta,nodo];                                                    (39)
     lista_rutas[ruta-1].Distancia     :=     lista_rutas[ruta-1].Distancia     +
matriz_dist[cab_ruta,nodo];                                                    (40)
     if lista_rutas[ruta-1].Tiempo < matriz_nodos[nodo,4] then                 (41)
        begin
             lista_rutas[ruta-1].Espera     :=     lista_rutas[ruta-1].Espera     +
(matriz_nodos[nodo,4] - lista_rutas[ruta-1].Tiempo);                           (42)
```

```
            lista_rutas[ruta-1].Tiempo       :=       matriz_nodos[nodo,4]       +
matriz_nodos[nodo,6];                                                          (43)
        end
     else
         lista_rutas[ruta-1].Tiempo      :=      lista_rutas[ruta-1].Tiempo      +
matriz_nodos[nodo,6];                                                          (44)
     lista_rutas[ruta-1].Carga        :=        lista_rutas[ruta-1].Carga        +
matriz_nodos[nodo,3];                                                          (45)
     SetLength(lista_rutas[ruta-1].Nodo,ultimo+1);
     lista_rutas[ruta-1].Nodo[ultimo] := nodo;                                 (46)
end;
```

Figura 17. Procedimiento de adición directa de un nodo a una ruta

Una vez asignado el nodo a una ruta en cuestión, se procede a actualizar todos los valores de la ruta. En primer lugar se calcula el tiempo estimado de llegada de la ruta al nodo en cuestión a través de la adición al tiempo de la ruta original, de la distancia a recorrer desde la cabeza de ruta hasta el nodo en cuestión (39). Igualmente se computa la distancia total recorrida por la ruta hasta el momento, y que equivale a la distancia recorrida acumulada, más la distancia recorrida hasta el nodo en cuestión (40). A continuación se procede al estudio de las condiciones de llegada de la ruta al nodo (41) a través del cómputo del momento estimado de llegada. Si el vehículo llega antes de la apertura del nodo, entonces es necesario computar una espera de ese vehículo hasta el momento de apertura del nodo (42), de manera que esta espera se acumula a las esperas totales provocadas en la ruta en cuestión. Si este es el caso, el tiempo de la ruta se actualiza con el valor del momento de apertura más el tiempo de servicio en ese nodo (43). El caso contrario implica una entrada directa en el nodo, cuando este permanece abierto, de manera que no es necesario realizar ninguna espera. Por ello el tiempo de la ruta se actualiza con el valor del tiempo de servicio, acumulado al anterior tiempo de la ruta (44). Por último se actualizan los valores de la carga de la ruta (45), y se indica cuál es el nuevo nodo que forma la cabeza de la ruta (46).

1.3.8.3 Procedimiento de generación de la lista de inserciones simples *L2* (Instrucción 32)

En el caso de que la holgura sea mayor que el mínimo tiempo de servicio, entonces es posible realizar alguna visita intermedia entre la cabeza de ruta y el nodo en cuestión, por lo que se procede a la generación de posibles candidatos a ser escalas intermedias entre estos dos nodos, tal y como establece la instrucción (32). Se trata de la generación de la lista *L2*, que contiene nodos no asignados de *L0*, que no distan mucho de la línea recta entre los dos nodos en estudio, y que además están abiertos cuando el vehículo pasa por ahí. Por ello se procede a la generación de la lista *L2* con posibles candidatos, tal y como se describe en el siguiente procedimiento.

```
Procedure Generar_Lista_Ins_Senc(nodo, ruta : integer);
var
   i, long, carga_final, cab_ruta: integer;
   holgura, holgura_2, aux_beta, t_final, aux_cierre : real;
   aux_reg : TnodoInsSenc;
begin
     SetLength(lista_ins_senc,0);
     for i:=1 to num_nodos do
```

```
         if not lista_asig[i] then                                                    (47)
         begin
              long := length(lista_rutas[ruta-1].Nodo);
              cab_ruta := lista_rutas[ruta-1].Nodo[long-1];
              aux_cierre  :=  matriz_nodos[i,5]  -  lista_rutas[ruta-1].Tiempo  -
matriz_dist[cab_ruta,i];                                                          (48)
              holgura    :=    matriz_nodos[nodo,5]    -    matriz_nodos[i,4]    -
matriz_nodos[i,6] - matriz_dist[i,nodo];                                          (49)
              holgura_2  :=  matriz_nodos[nodo,5]  -  lista_rutas[ruta-1].Tiempo  -
matriz_dist[cab_ruta,i] - matriz_nodos[i,6] - matriz_dist[i,nodo];                (50)
              aux_beta     :=     (beta     *     matriz_dist[cab_ruta,nodo])     -
matriz_dist[cab_ruta,i] - matriz_dist[i,nodo];                                    (51)
              t_final  :=  lista_rutas[ruta-1].Tiempo  +  matriz_dist[cab_ruta,i]  +
matriz_nodos[i,6]     +     matriz_dist[i,nodo]     +     matriz_nodos[nodo,6]     +
matriz_dist[nodo,0];                                                              (52)
              carga_final  :=  lista_rutas[ruta-1].Carga  +  matriz_nodos[i,3]  +
matriz_nodos[nodo,3];                                                             (53)
              if (aux_cierre >= 0) and (holgura >= 0) and (holgura_2 >= 0) and
(aux_beta >= 0) and (t_final <= matriz_nodos[0,5]) and (carga_final <= carga_cam)
               then                                                               (54)
                 begin
                      long := length(lista_ins_senc);
                      SetLength(lista_ins_senc,long+1);
                      lista_ins_senc[long].Nodo := i;
                      lista_ins_senc[long].Holgura := holgura_2;
                 end;
         end;
     for i:=1 to length(lista_ins_senc)-1 do
         if (lista_ins_senc[i].Holgura) > (lista_ins_senc[0].Holgura) then
            begin
                 aux_reg := lista_ins_senc[0];
                 lista_ins_senc[0] := lista_ins_senc[i];
                 lista_ins_senc[i] := aux_reg;
            end;
end;
```

Figura 18. Procedimiento de generación de la lista de *L2* de posibles escalas simples

En el procedimiento anterior se describe la generación de la lista *L2* de nodos candidatos a ser escala intermedia. En primer lugar (47) se seleccionan un conjunto de nodos de *L0* que no estén asignados. Para ello es necesario comprobar que el nodo *i* en cuestión ha de estar abierto antes que el nodo crítico, para lo que se establece la restricción (49) que comprueba que los nodos estén abiertos cuando el vehículo pase por los mismos. También han de estar abiertos en un momento tal que su servicio y posterior desplazamiento desde el nodo insertado al nodo añadido permiten llegar antes del momento de cierre del nodo a ser añadido. Igualmente se ha de respetar la restricción de llegada estimada que establece que los nodos candidatos a ser insertados han de ser tales que el tiempo de la ruta, incrementado en los desplazamientos desde la cabeza de ruta, hasta el nodo a ser insertado, y desde este hasta el nodo crítico en estudio, y añadido el tiempo de servicio del nodo a ser insertado, entonces permite llegar antes del cierre del nodo crítico (50), y además no ha de suponer un gran desplazamiento (51). Por último (48) contempla que el nodo a ser insertado no esté cerrado cuando llegue el vehículo a realizar su servicio. Finalmente, en las restricciones (52) y (53) también se contempla que la visita de este nodo intermedio no genere retrasos globales en la ruta que impidan llegar al depósito central antes del cierre, y que tampoco supongan una violación de las restricciones de carga del camión. Si y solamente si se cumplen todas y cada una de estas

restricciones, entonces se seleccionan los nodos a ser insertados (54) en la lista *L2*. Estos nodos se ordenan en función de la holgura resultante tras realizar la visita intermedia, y se selecciona aquel que proporciona mayor holgura. Esta selección se realiza al final de este procedimiento, y el resultado del mismo es el conjunto de nodos que integran *L2*, ordenados de mayor a menor holgura.

1.3.8.4 Asignación directa (Instrucción 33)

En el caso en el que la lista *L2* se encuentre vacía entonces se procede a la adición del nodo crítico seleccionado directamente a la ruta a la que fue previamente según los procedimientos (30) y (31), calculando la correspondiente espera.

1.3.8.5 Recálculo de holguras (Instrucción 34)

Sin embargo, si la lista *L2* no está vacía, entonces (34) se vuelven a comprobar las holguras resultantes en el caso de poder establecer dos inserciones simultáneas de nodos. De esta manera sabremos si es posible insertar dos nodos, o no. Para ello se desarrolla una función de comprobación del tamaño de las holguras resultantes de dos inserciones y que se expone a continuación. Para ello se establece una variable booleana que tomará el valor de "*verdadero*" si existen holguras resultantes mayores que el menor tiempo de servicio, y tomará el valor "*falso*", si las holguras resultantes son todas menores que el menor tiempo de servicio, de manera que no sería posible ningún tipo de inserción doble.

```
function Comprobar_Holguras_L2 : boolean;
var
   i, menor : integer;
   aux : boolean;
begin
     menor := Hallar_Minimo_T_Serv;
     aux := false;
     for i:=0 to length(lista_ins_senc)-1 do
         if lista_ins_senc[i].Holgura >= menor then                          (55)
            aux := true;
     if length(lista_ins_senc) < 2 then                                      (56)
        aux := false;
     result := aux;
end;
```

Figura 19. Comprobación de posible inserción doble

1.3.8.6 Generación de la lista *L3* de inserciones dobles (Instrucción 35)

```
procedure Generar_Lista_Ins_Doble(nodo, ruta : integer);
var
   k, l, nodo_k, nodo_l, long_senc, long_doble, cab_ruta, long : integer;
   holgura_k, holgura_l, holgura_n, holgura_final : real;
   aux_1, aux_2, aux_3, aux_k, aux_l, aux_beta, aux_carga : real;
   aux_reg : TNodoInsDoble;
begin
     SetLength(lista_ins_doble,0);
     long_senc := length(lista_ins_senc);
     for k:=0 to long_senc-1 do
         for l:=0 to long_senc-1 do
             if k<>l then                                                    (57)
                begin
                     long := length(lista_rutas[ruta-1].Nodo);
                     nodo_k := lista_ins_senc[k].Nodo;
```

```
                    nodo_l := lista_ins_senc[l].Nodo;
                    cab_ruta := lista_rutas[ruta-1].Nodo[long-1];
                    aux_1         :=         lista_rutas[ruta-1].Tiempo         +
matriz_dist[cab_ruta,nodo_k];                                                    (58)
                    aux_2 := matriz_nodos[nodo_k,6] + matriz_dist[nodo_k,nodo_l];
                                                                                 (59)
                    aux_3  :=  matriz_nodos[nodo_l,6]  +  matriz_dist[nodo_l,nodo];
                                                                                 (60)
                    holgura_k := matriz_nodos[nodo_k,5] - aux_1;                 (58)
                    holgura_l := matriz_nodos[nodo_l,5] - aux_1 - aux_2;         (59)
                    holgura_n  :=  matriz_nodos[nodo,5]  -  aux_1  -  aux_2  -  aux_3;
                                                                                 (60)
                    holgura_final := matriz_nodos[0,5] - aux_1 - aux_2 - aux_3 -
matriz_nodos[nodo,6] - matriz_dist[nodo,0]; (62)
                    aux_k  :=  matriz_nodos[nodo_l,5]  -  matriz_nodos[nodo_k,4]  -
aux_2;
                    aux_l  :=  matriz_nodos[nodo,5]  -  matriz_nodos[nodo_l,4]  -
aux_3;
                    aux_beta  :=  (gamma  *  beta  *  matriz_dist[cab_ruta,nodo])  -
matriz_dist[cab_ruta,nodo_k]         -         matriz_dist[nodo_k,nodo_l]         -
matriz_dist[nodo_l,nodo];                                                        (61)
                    aux_carga        :=        lista_rutas[ruta-1].Carga        +
matriz_nodos[nodo_k,3] + matriz_nodos[nodo_l,3] + matriz_nodos[nodo,3];       (63)
                    if (holgura_k >= 0) and (holgura_l >= 0) and (holgura_n >= 0)
and (holgura_final >= 0)
                      and (aux_k >= 0) and (aux_l >= 0) and (aux_beta >= 0) and
(carga_cam >= aux_carga) then
                    begin
                         long_doble := length(lista_ins_doble);
                         SetLength(lista_ins_doble,long_doble+1);
                         lista_ins_doble[long_doble].NodoIni := nodo_k;
                         lista_ins_doble[long_doble].NodoFin := nodo_l;
                         lista_ins_doble[long_doble].Holgura := holgura_n;(64)
                    end;
               end;
     long_doble := length(lista_ins_doble);
     for k:=1 to long_doble-1 do
         if lista_ins_doble[k].Holgura > lista_ins_doble[0].Holgura then
            begin
                 aux_reg := lista_ins_doble[0];
                 lista_ins_doble[0] := lista_ins_doble[k];
                 lista_ins_doble[k] := aux_reg;
            end;
end;
```

Figura 20. Generación de la lista L3 de inserciones dobles

En el caso de que sí sea posible, entonces procedemos al cálculo de la lista *L3* de inserción doble (35). Para ello se buscan los nodos que cumplan las restricciones para formar parte de *L3*, y que van a provenir de *L2*.

En este procedimiento se presentan los nodos a ser candidatos de inserciones dobles. Estas inserciones dobles se realizan a través de los nodos *k* y *l* de *L2* de forma que $k \neq l$ (57) tal y como se presenta en este algoritmo. Para ello todo par de nodos *k* y *l* incluidos en *L2*, siempre se ordenan de esta manera, de forma que las restricciones se fundamentan en este orden. Estas restricciones comienzan sobre el nodo *k*. En este caso se plantean las restricciones temporales del problema de la siguiente manera.

En (58) se presentan las restricciones temporales que afectan al nodo *k* en cuestión, y que implica que cuando el vehículo llegue al nodo *k* éste ha de estar abierto. Dicho de otra forma, para el nodo *k* se establece que: $tr_z + t_{CRz\,k} \leq tc_k$

En este sentido la restricción implica que el nodo *k* elegido ha de ser tal que su momento de cierre ha de ser posterior al tiempo total acumulado en la ruta, más el tiempo necesario para llegar a este nodo. Por ello en (41) se establece la inecuación correspondiente computándose una variable de holgura.

Para el nodo *l* en cuestión se establece que: $tr_z + t_{CRz\,k} + ts_k + t_{kl} \leq tc_l$

En este caso la restricción indica que el nodo *l* escogido ha de ser tal que su momento de cierre ha de permitir el desplazamiento del vehículo de la cabeza de ruta, hasta el nodo *k*, hacer el servicio en el nodo *k*, y desplazarse hasta el nodo *l*, de manera que se llegue al nodo *l* antes de que este cierre. Igual que en el caso anterior se integra la correspondiente variable de holgura (59).

Por último para el nodo *i*, también se establecen que: $tr_z + t_{CRz\,k} + ts_k + t_{kl} + ts_l + t_{l\,i} \leq tc_i$

En este caso la restricción (60) indica que la selección de ambos nodos ha de ser tal que el tiempo de llegada al nodo en consideración sea anterior a su momento de cierre, permitiendo por tanto el desplazamiento desde la cabeza de ruta, hasta el nodo *k*, realizar el servicio en el nodo *k*, desplazarse desde allí hasta el nodo *l*, realizar el servicio en el nodo *l*, y finalmente desplazarse hasta el nodo *i* en consideración. Igual que en los dos casos anteriores, se computa una nueva variable de holgura para este nodo *i*, que se tendrá en cuenta para la selección final de un par de nodos *(k, l),* y que deberán de proporcionar aquella nueva holgura que resulte ser máxima, o lo que es lo mismo, que los nodos *k* y *l* seleccionados se encuentren lo más enfilados posible entre la cabeza de ruta correspondiente y el nodo *i* en consideración, respetando igualmente la restricción de desplazamiento geográfico máximo impuesto por β y γ (61) y que se representa por la siguiente inecuación: $t_{CRz\,k} + t_{kl} + t_{li} \leq \gamma \cdot \beta \cdot t_{CRz\,i}$

Dicho de otra forma, el desplazamiento total causado por la inserción de los nodos *k* y *l* ha de ser siempre menor o igual que $\gamma \times \beta$ veces la distancia en línea recta entre la cabeza de ruta y el nodo a añadir. Cabría plantearse que estos nodos *k* y *l* podrían no estar abiertos durante su selección, y por ello también se incluyen las correspondientes restricciones relativas a `aux_k` y `aux_l`.

Igualmente se establece la restricción correspondiente a la no violación de los tiempos de cierre del depósito si es que se realizan las visitas intermedias, lo que queda plasmado en la restricción (62) que indica que todos los recorridos realizados, incluyendo el servicio del nodo *i* en consideración, y el hipotético regreso desde *i* hasta el depósito, deberían de ocurrir antes del cierre. Por lo que: $tr_z + t_{CRz\,k} + ts_k + t_{kl} + ts_l + t_{l\,i} + ts_i + t_{i0} + te_{ki} \leq tc_0$

También se establece la restricción de cargas oportuna, que indica que la inserción de los nodos *k* y *l* en consideración ha de respetar la carga máxima del vehículo, lo que queda plasmado en la restricción (63), de forma que la carga acumulada de la ruta permita la inserción de las cargas de los nodos *k, l*, e *i*, sin violar las restricciones de carga del vehículo.

En este caso si se cumplen todas y cada una de las restricciones expresadas anteriormente, entonces los nodos *k* y *l* pasan a formar parte de los nodos a ser insertados conjuntamente entre la cabeza de ruta y el nodo *i* en estudio, computándose para cada par de nodos una nueva holgura, tal y como aparece en el la restricción (60), que indica que esta holgura ha de ser mayor o igual que *0*, indicando que se ha de llegar antes o justo en el momento de cierre del nodo en consideración, después de realizar la inserción. Por eso la elección del par de nodos *k* y *l* ha de ser aquella que maximice esta holgura. Consecuentemente la lista *L3* se ordena de forma ascendente de acuerdo con la nueva holgura resultante, tal y como se define en (64), y a partir de ahí se establece las instrucciones oportunas para que dicha lista aparezca ordenada de forma ascendente, ocupando el lugar *[0]* en la lista aquél par *(k, l)* que proporciona la máxima holgura.

1.3.8.7 Procedimiento de inserción doble (Instrucción 36)

A continuación, en (36), si existen algún par de nodos en la lista *L3*, entonces se procede a la inserción doble de aquel par que maximiza la nueva holgura resultante. Para ello se realizan tres adiciones simples simultáneamente, según el procedimiento descrito en (30) y (31). Primero se añade el nodo *k* a la ruta, recalculando todos los valores según el procedimiento descrito anteriormente. Una vez añadido, se cambia su variable booleana a "*asignado*". A Continuación se procede a la adición del nodo *l* en consideración, e igualmente se contempla su cambio a la posición de "*asignado*", y por último se añade finalmente el nodo *i*, que es el que puede ser añadido con o sin espera, computada también por el procedimiento anteriormente descrito en (30) y (31). Este nodo ya había sido considerado como asignado en su inclusión en *L1*, tal y como se contempla previamente.

1.3.8.8 Procedimiento de inserción simple (Instrucciones 37 y 38)

En el caso en el que no exista ningún par que cumpla todas estas restricciones entonces se procede a la adición del nodo anteriormente considerado en *L2*, tal y como figura en (37), y después de cambiar su posición a asignado, se procede a añadir el nodo crítico en cuestión.

1.3.8.9 Procedimiento de finalización de las rutas (Instrucción 8)

El algoritmo seguirá trabajando hasta que finalice los nodos no asignados de *L0*. Cuando estos finalicen, entonces el algoritmo provoca artificialmente que todos los nodos regresen al origen, computando estos últimos desplazamientos, y proporcionando finalmente los datos de la solución.

En este procedimiento se añade finalmente el retorno de todos los vehículos al depósito central, dentro de los plazos establecidos para el mismo. Cuando ya no quedan más nodos por asignar, se añade el depósito a todas las rutas (65), de manera que el último nodo visitado por las mismas es el nodo *0*. Igualmente se procede al cálculo de los tiempos finales de cada una de las rutas actualizando el dato acumulado de cada una de las mismas con el tiempo necesario para realizar el retorno al depósito central (66). Lo mismo se realiza con la distancia recorrida por cada uno de los vehículos (67).

```
procedure Finalizar_Rutas;
var
   i, long : integer;
begin
     for i:=1 to num_rutas do
     begin
          long := length(lista_rutas[i-1].Nodo);
          SetLength(lista_rutas[i-1].Nodo,long+1);
          lista_rutas[i-1].Nodo[long] := 0;                                    (65)
          lista_rutas[i-1].Tiempo        :=        lista_rutas[i-1].Tiempo        +
matriz_dist[lista_rutas[i-1].Nodo[long-1],0];                                  (66)
          lista_rutas[i-1].Distancia      :=      lista_rutas[i-1].Distancia      +
matriz_dist[lista_rutas[i-1].Nodo[long-1],0];                                  (67)
     end;
end;
```

Figura 21. Procedimiento de regreso de vehículos al depósito central

De esta manera finalizan todos los procedimientos, funciones e instrucciones relativas a la ejecución del algoritmo propiamente dicho. Si bien es necesario contemplar todavía las instrucciones relativas al desarrollo de la interfaz, así como a las funciones de la misma que *lanzan* los procedimientos de cálculo incluidos en este apartado. Para ello en el siguiente punto se contemplan todos estos procedimientos relativos al desarrollo de la interfaz y de las funciones del software aquí desarrolladas. Igualmente en el siguiente apartado se presentan las instrucciones de manejo de la aplicación a modo de un *manual de usuario*, aunque es necesario mantener en todo momento la consideración de que se trata de un software desarrollado con fines relativos a la investigación, dado que no se trata de un software destinado al uso empresarial.

2 Interfaz de la aplicación

En las siguientes líneas se describe detalladamente el conjunto de instrucciones que hacen referencia al funcionamiento de la aplicación en conjunto, y en especial al manejo de la interfaz.

2.1 Presentación de la interfaz de usuario

Una vez ejecutada la aplicación, ésta genera la interfaz de la misma, desde la que el usuario puede introducir todos los datos necesarios del problema, así como establecer los parámetros de control para posteriormente recibir todos los valores de la solución encontrada por el algoritmo.

Esta interfaz contempla dos áreas básicas. En el lado izquierdo de la interfaz se aportarán todos los datos requeridos por el programa para la aplicación de las reglas de decisión, y en el lado derecho se proporcionarán las soluciones al usuario.

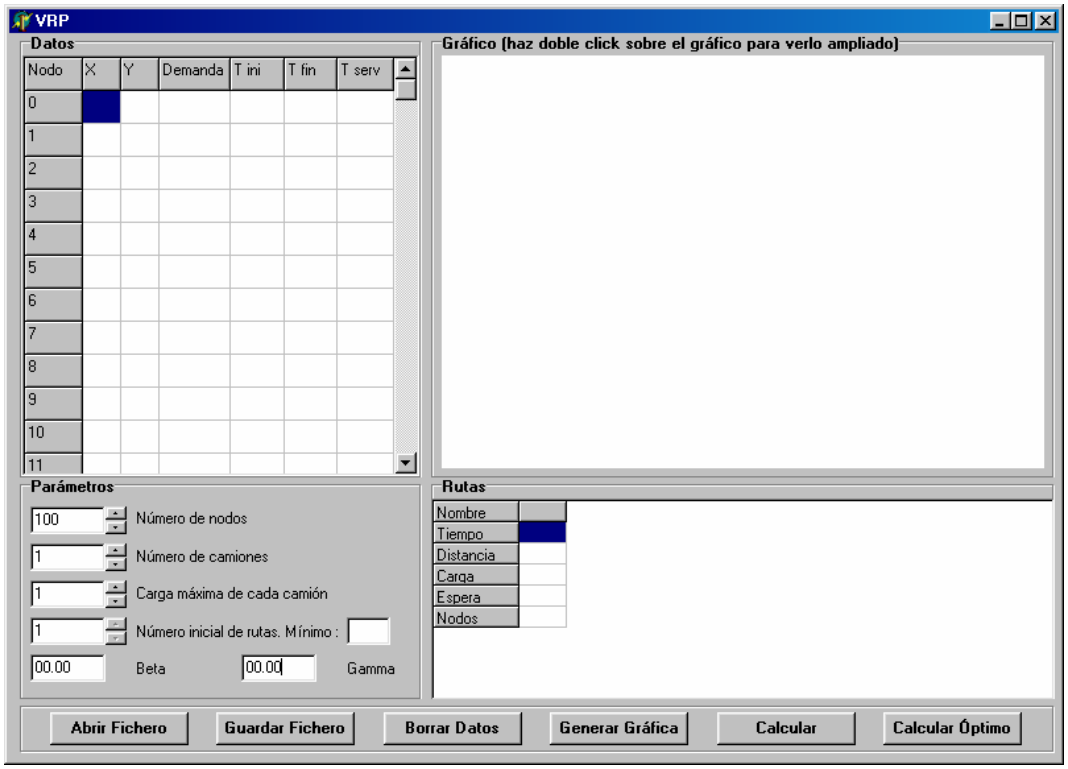


Figura 22. Imagen de la Interfaz de la aplicación

2.1.1 Datos a definir

Estos datos a ser introducidos por el usuario son los siguientes:

- **Número de nodos de los que consta el problema:** Esta variable del problema hace referencia al número total de clientes que han de ser servidos desde el depósito central, y que puede oscilar según los problemas que se traten de resolver. En los problemas tipo aportados en la literatura generalmente se contemplan problemas de 100 nodos con diferentes distribuciones de los mismos. Sin embargo también se han desarrollado versiones más simplificadas con 50 y 25 nodos de estos mismos problemas tipo. Por ello en este trabajo se ha optado por establecer la posibilidad de introducir cantidades variables de clientes a ser servidos por la flota de camiones disponible, para poder afrontar cualquiera de los problemas tipo de la literatura, así como para garantizar una mayor adaptabilidad de los fundamentos del algoritmo a la situación real del problema.

  Estos nodos cuentan además con el nodo central o depósito desde el que parten y al que han de llegar todos y cada uno de los camiones. Este nodo se identifica con el nodo *0*, por ello el número total de nodos de todo problema será el número de clientes *+1*.

  La tabla en cuestión contiene toda la información necesaria de cada nodo, incluyendo los siguientes datos:

- **Nodo:** La columna nodo identifica a todos y cada uno de los nodos existentes en el problema, siendo *0* el nombre del nodo que hace de depósito, y *1.... N* los nombres de los siguientes nodos. En una planificación real, los nombres de los nodos serían propiamente nombres de clientes o de ciudades.
- **Coordenada *X*:** Es la coordenada relativa al eje horizontal, y que se podría identificar con las coordenadas geográficas del nodo en cuestión. En este caso se correspondería con la "*longitud*" Mide la distancia horizontal con respecto al eje de ordenadas.
- **Coordenada *Y*:** Analógicamente representa la coordenada relativa al eje vertical, y por ello se identificaría con la "*latitud*". Mide la distancia vertical con respecto al eje de abscisas.
- **Demanda:** La demanda hace referencia a la cantidad de producto demandada por el cliente en cuestión. Se podría identificar con los kilogramos, litros, o cualquier otra unidad de medida que se ha de servir, o bien recoger, si aplicamos el criterio de recogidas. Si bien en este trabajo no se ha contemplado la coexistencia de recogidas y entregas simultáneas.
- **Tiempo de apertura:** El tiempo o momento de apertura hace referencia al momento temporal en el que el nodo en cuestión abre para la entrega de mercancías, y a partir del cual ha de entrar un vehículo para realizar el servicio.
- **Tiempo de cierre:** Es el momento temporal en el que el nodo en cuestión cierra sus puertas y a partir del cuál no podría ser servido por ningún vehículo.
- **Horizonte total de planificación:** El nodo central o depósito marca el horizonte temporal en el que abren y cierran todos los nodos.

- **Tiempo de servicio:** Por último, se establece la última variable que es el tiempo de servicio requerido por el vehículo para realizar la entrega en el nodo correspondiente.

Igualmente se establecen como datos el **número de vehículos** disponible para realizar la planificación de las rutas, así como la **capacidad máxima** de cada uno de los vehículos.

Estos datos han de ser siempre aportados por el usuario del algoritmo, y constituyen la fuente principal de información del mismo. En los problemas tipo, siempre se cuenta con esta información, que generalmente está configurada en archivos con formatos `.txt` para su importación desde cualquier aplicación informática.

2.1.2 Parámetros a seleccionar

En el cuadrante inferior izquierda se contemplan los diferentes parámetros de las que consta el problema. Estos parámetros son definidos por el usuario para conseguir diferentes soluciones.

- **Número inicial de rutas:** Este parámetro obedece a la necesidad de planificar desde el comienzo del horizonte un número determinado de rutas, a partir del cuál se irá incrementando en función de las necesidades del problema. Se corresponde con el número total de nodos semilla con los que el usuario decide iniciar la planificación. Los dos extremos consistirían en comenzar con un número muy reducido de rutas, de forma que la posible solución fuese cercana al óptimo desde el principio, aunque esto resultase en que las siguientes rutas que se crearían por necesidades de la planificación, no hubieran tenido en cuenta nodos ya asignados a otras rutas. Es decir, que las primeras rutas en ser seleccionadas escogerían nodos que no resultarían demasiado buenos, y que por ello ya no se contemplarían en las siguientes rutas, pudiendo ser mejores elecciones para estas últimas. La segunda opción consistirían en comenzar con un número elevado de rutas, de manera que se perjudicaría la solución global debido a la existencia de un gran número de rutas, aunque dentro de cada ruta, los nodos considerados sí serían buenas elecciones.
- **Parámetro β:** Se trata del parámetro penalizador de desplazamientos elevados en las posibles inserciones simples de nodos cada vez que se considera un nodo crítico.
- **Parámetro γ:** Consiste en el parámetro penalizador de desplazamientos en las posibles inserciones dobles de nodos entre el nodo cabeza de ruta y el nodo crítico anterior. Recordemos que se considera conjuntamente con el parámetro β.

2.1.3 Rutas generadas

La parte derecha de la interfaz de usuario proporciona los outputs del algoritmo y que son propiamente las rutas, con sus respectivos programas de visitas. Se contempla la solución desde dos puntos de vista complementarios, por un lado se proporciona al usuario una visualización en el espacio de las diferentes rutas de las que consta la solución del problema, y por otro lado se proporcionan los diferente programas de visitas con los nodos de cada una de las rutas, así como los tiempos de espera de cada una.

2.1.4 Gráficos de las rutas

Todas las rutas parten del depósito central, y realizan las correspondientes visitas. Se muestran en diferentes colores las distintas rutas. Igualmente es posible obtener una vez resuelto el problema, una ampliación del gráfico de las rutas en consideración para una mejor interpretación de los resultados por parte del usuario. En esta ampliación es posible identificar los nodos iniciales de cada una de las rutas, de las que se aporta el detalle en la lista incluida en el punto anterior. De esta manera, y combinando la utilización de la lista de rutas, y del gráfico, es posible interpretar con detalle todas las rutas.

2.2 Controles de la aplicación

2.2.1 "Abrir Fichero"

Este procedimiento que sigue a continuación contiene las instrucciones relativas a la importación de datos de un fichero *.txt* a la tabla de datos de entrada de la aplicación. En este sentido se ha respetado el formato en el que se recogen de la literatura todos los problemas tipo y que se encuentran en http://www.idsia.ch/~luca/macs-vrptw/problems/ De estos ficheros se extrae la información relativa a las siguientes variables:

- Nombre del problema tipo, tal y como se definían en el capítulo del Estado de la Cuestión.
- Número de vehículos disponibles para realizar la planificación de las rutas
- Capacidad máxima de estos vehículos
- Nombre de cada uno de los clientes a tener en cuenta
- Coordenada *X* de la localización del cliente
- Coordenada *Y* de la localización del cliente
- Demanda del cliente
- Apertura del cliente
- Cierre del cliente
- Tiempo de servicio necesario en ese cliente.

Una vez recogidos los datos del fichero en cuestión se vuelcan en la aplicación, en sus ubicaciones correspondientes para proceder a la generación de la representación gráfico así como del propio cálculo de los resultados.

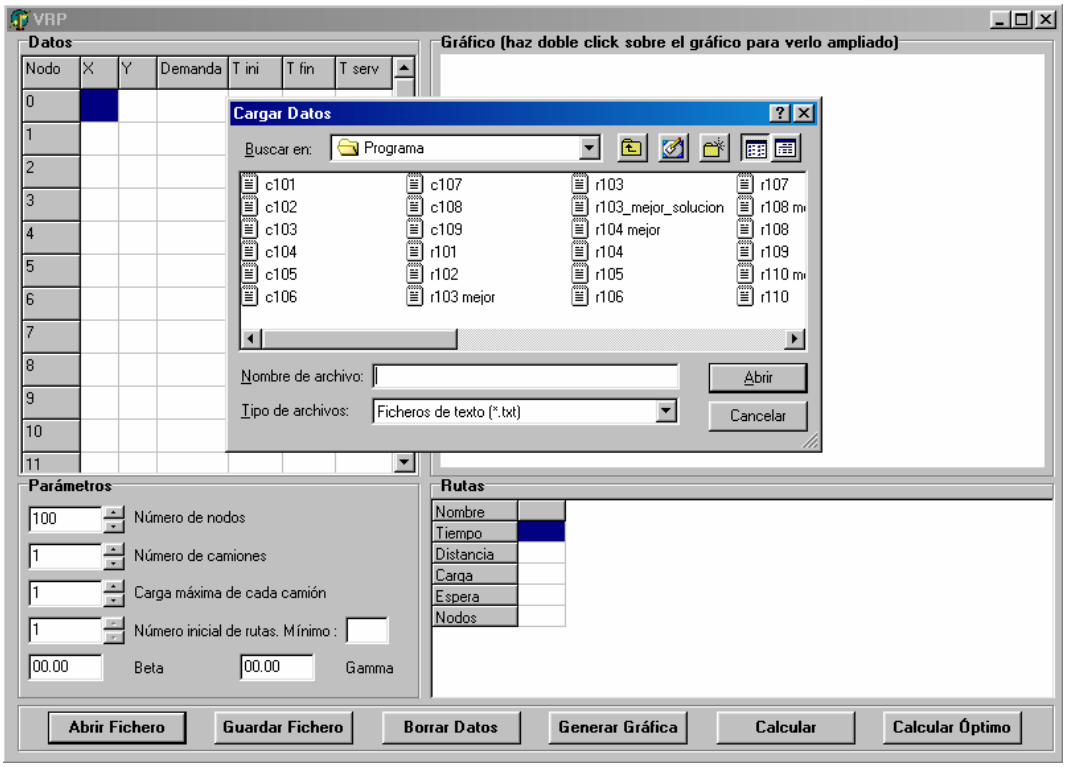


Figura 23. Procedimiento de "Abrir Fichero"

r103 - Bloc de notas

Archivo Edición Buscar Ayuda

R103

VEHICLE
NUMBER CAPACITY
25 200

CUSTOMER

CUST NO.	XCOORD.	YCOORD.	DEMAND	READY TIME	DUE DATE	SERVICE TIME
0	35	35	0	0	230	0
1	41	49	10	0	204	10
2	35	17	7	0	202	10
3	55	45	13	0	197	10
4	55	20	19	149	159	10
5	15	30	26	0	199	10
6	25	30	3	99	109	10
7	20	50	5	0	198	10
8	10	43	9	95	105	10
9	55	60	16	97	107	10
10	30	60	16	124	134	10
11	20	65	12	67	77	10
12	50	35	19	0	205	10
13	30	25	23	159	169	10
14	15	10	20	0	187	10
15	30	5	8	61	71	10
16	10	20	19	0	190	10
17	5	30	2	157	167	10
18	20	40	12	0	204	10
19	15	60	17	0	187	10
20	45	65	9	0	188	10
21	45	20	11	0	201	10
22	45	10	18	97	107	10
23	55	5	29	68	78	10
24	65	35	3	0	190	10
25	65	20	6	172	182	10

Figura 24. Formato estándar de los ficheros de los casos de Solomon (1987)

2.2.2 "Guardar Fichero"

Se ha contemplado la posibilidad de guardar los datos en ficheros de texto `.txt` para facilitar una vez más las posibilidades de exportar estos datos a otras aplicaciones desde las que poder realizar tratamientos sobre los mismos.

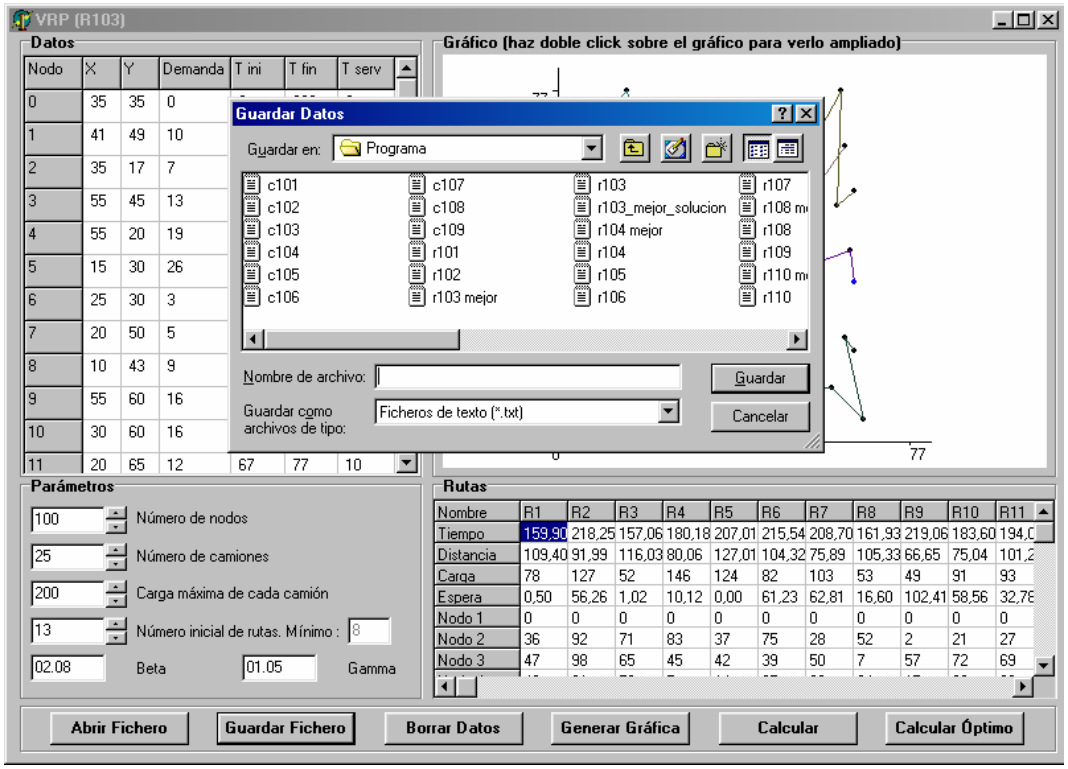


Figura 25. Procedimiento de "Guardar Fichero".

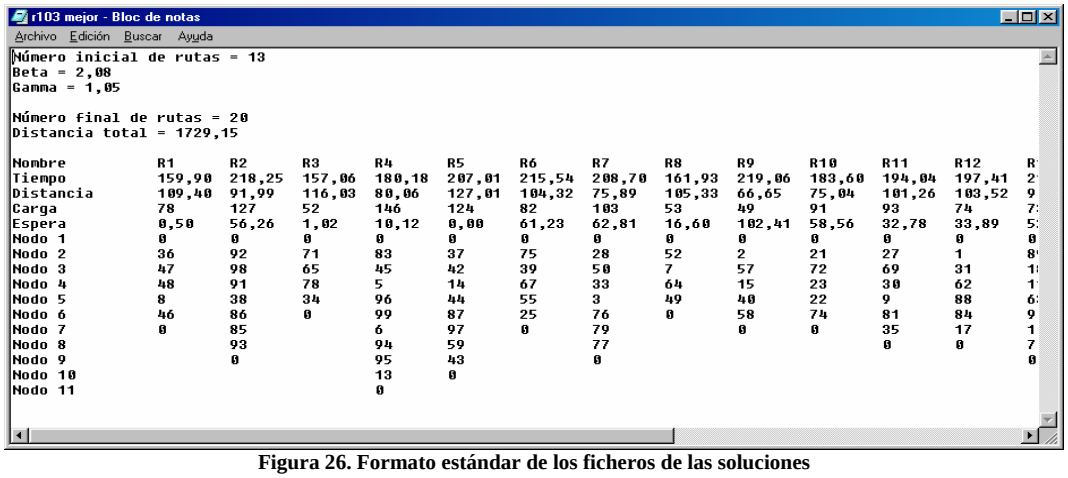


Figura 26. Formato estándar de los ficheros de las soluciones

2.2.3 "Borrar Datos"

En el procedimiento que sigue a continuación, se asocia al botón "*Borrar*", la eliminación de todos los datos del problema, tanto los relativos a la solución, como a los propios datos del problema en cuestión. En este procedimiento se inicializan todas las variables del problema, por lo que es necesario haber guardado la solución previamente, dado que sino ésta se perderá con el borrado.

2.2.4 "Generar Gráfica"

En este procedimiento asociado al botón "*Generar gráfica*", el algoritmo realiza la representación de los datos incluidos en la interfaz. Esta representación se realiza en el cuadrante superior derecho, y en el se representan tanto el depósito central, en color rojo, como el resto de los puntos, en color negro.

2.2.5 "Calcular"

El algoritmo proporciona la solución en un cuadro de diálogo en la pantalla en el que se indican los resultados de la solución. A continuación se vuelcan en la tabla de rutas, posicionada en el cuadrante inferior derecha todos los resultados de cada una de las rutas. Estos resultados son devuelto por el motor de cálculo del algoritmo. Por último, se grafican todos los nodos componentes de la solución . Para ello, se utilizan tres colores, el rojo para el depósito central, el azul para los nodos comienzo de cada una de las rutas integrantes de la solución, y el negro para el resto de los nodos. Igualmente se grafican las rutas, a través de la unión de cada dos visitas consecutivas en una misma ruta. Para ello cada una de las rutas se dibuja en un color escogido al azar, de manera que es posible diferenciarlas visualmente en el gráfico.

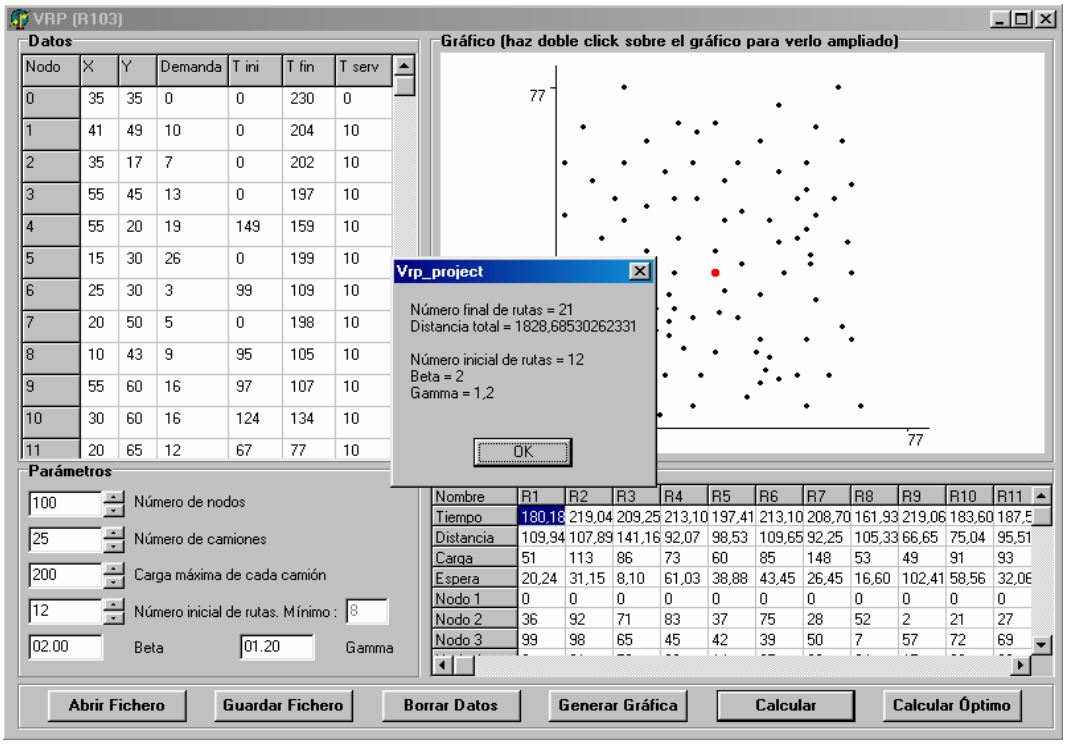


Figura 27. Ventana con los datos relativos a la solución

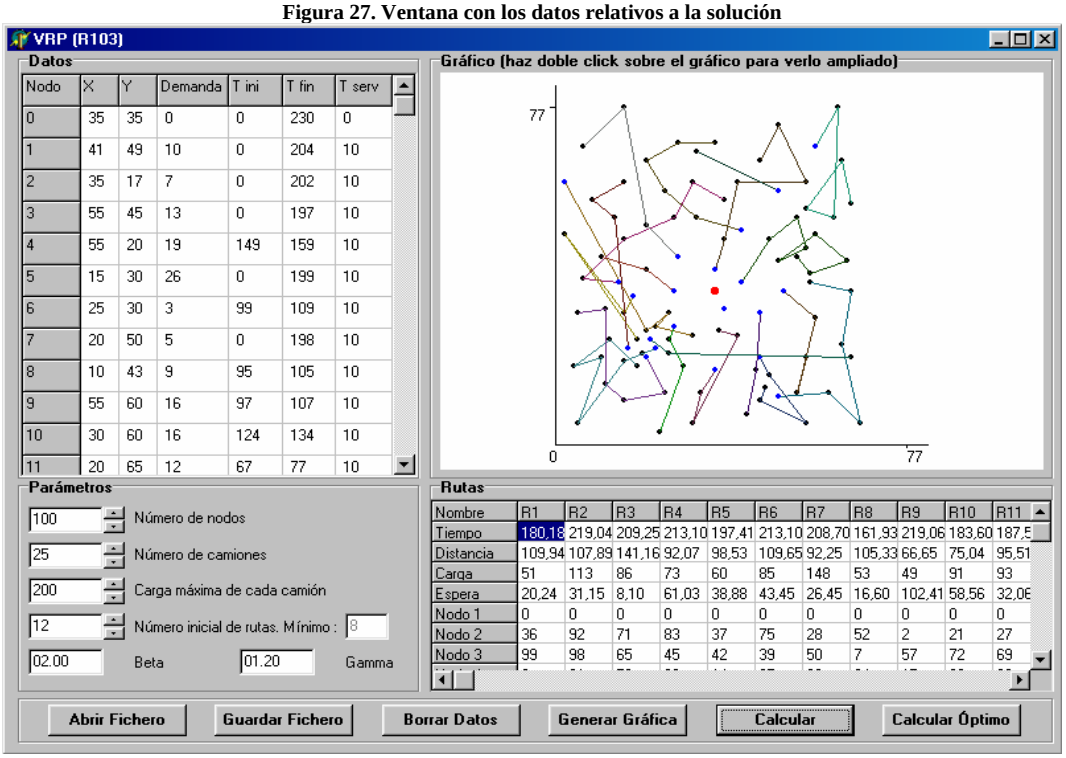


Figura 28. Representación en la interfaz de los datos integrantes de la solución

2.2.6 "Calcular Óptimo"

El algoritmo procede de igual manera que en el caso anterior pero comprobando todas las posibles combinaciones de los parámetros integrantes de los datos del problema, y proporcionando finalmente la mejor solución obtenida. Este procedimiento suele conllevar largos períodos de computación.

Anexo III

Datos de las gráficas para el ejemplo R103

	Valores de *R*																			
Valores de *β*	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	2168	1993	2065	2083	2122	2120	2076	2057	2081	2060	2103	2233	2148	2052	2124	2029	2163	2105	2215	2204
1,05	2047	1941	1931	1932	1983	1941	1911	1803	1929	1966	1927	2228	1982	1868	1914	1870	1941	1905	2041	2031
1,1	2034	1940	1906	1813	1960	1940	1888	1861	1837	1928	1920	1967	1972	1827	1828	1827	1845	1945	2055	2014
1,15	2034	1940	1870	1813	1820	1939	1837	1800	1819	1929	1924	2052	1960	1839	1815	1833	1851	1936	2060	2008
1,2	1953	1907	1849	1803	1800	1867	1894	1807	1811	1932	1930	1982	1999	1822	1923	1878	1908	1894	2087	1976
1,25	1925	1885	1843	1828	1803	1848	1899	1805	1825	1915	1857	2032	1993	1820	1847	1776	1849	1939	2063	1993
1,3	1927	1876	1845	1830	1804	1774	1907	1804	1792	1916	1905	2034	1995	1820	1872	1770	1849	1939	2109	1992
1,35	1927	1876	1853	1830	1804	1789	1935	1812	1801	1940	1905	2016	1995	1824	1872	1778	1849	1939	2109	1992
1,4	1869	1876	1853	1851	1804	1789	1935	1836	1818	1940	1913	2027	2026	1832	1927	1772	1856	1947	2117	1999
1,45	1891	1887	1853	1851	1804	1841	1935	1778	1818	1940	1913	2045	2026	1832	1927	1783	1861	1923	2078	1967
1,5	1895	1854	1853	1854	1807	1799	1928	1796	1830	1928	1919	1952	2026	1832	1927	1781	1895	1923	2044	1967
1,55	1895	1854	1894	1854	1830	1816	1864	1813	1830	1886	1899	1952	1964	1858	1909	1781	1862	1903	2050	1953
1,6	1833	1854	1894	1827	1902	1816	1864	1813	1923	1899	1954	1960	1964	1888	1909	1784	1862	1903	2050	1953
1,65	1883	1874	1894	1822	1923	1795	1864	1803	1880	1899	1954	1988	1975	1888	1907	1784	1855	1903	2050	1953
1,7	1883	1874	1894	1822	1923	1804	1864	1802	1880	1872	1954	1988	1993	1888	1907	1806	1855	1903	2047	1942
1,75	1883	1874	1894	1822	1923	1804	1864	1797	1877	1857	1956	2011	1988	1880	1911	1806	1843	1905	2028	1942
1,8	1883	1885	1894	1830	1935	1785	1864	1814	1875	1857	1883	2011	1920	1854	1911	1754	1843	1905	2028	1942
1,85	1871	1885	1894	1874	1935	1805	1864	1814	1875	1857	1883	1973	1920	1917	1934	1768	1912	1942	2075	1946
1,9	1871	1902	1910	1839	1935	1807	1933	1814	1875	1857	1982	1962	1931	1856	1944	1814	1912	1954	2070	1955
1,95	1900	1963	1910	1839	1891	1834	1933	1893	1875	1857	1982	1950	1931	1856	1944	1829	1912	1954	2070	2016
2	1914	1997	1912	1842	1891	1838	1934	1897	1875	1857	1923	1948	1933	1865	1949	1829	1912	1954	2070	2016
2,05	1914	1997	1912	1842	1891	1879	1990	1897	1875	1857	1923	1948	1933	1865	1949	1829	1912	1954	2070	2016
2,1	1903	1950	1920	1859	1942	1853	1991	1897	1856	1857	1923	1948	1931	1890	1949	1814	1915	1954	2070	2030
2,15	1903	1950	1920	1859	1873	1854	1991	1897	1829	1857	1931	1948	1931	1890	1949	1794	1950	1962	2063	1952
2,2	1903	1950	1998	1859	1873	1832	1991	1894	1829	1889	2005	1959	1940	1908	1927	1794	1950	1952	2077	1952
2,25	1903	1950	1998	1863	1873	1832	1991	1894	1829	1914	2005	1959	1940	1908	1927	1839	1908	1952	2077	1952
2,3	1903	1950	1998	1863	1878	1832	1984	1894	1841	1904	2011	1959	1983	1908	1927	1813	1869	1952	2077	1949
2,35	1908	1950	1998	1863	1878	1832	1984	1909	1841	1890	2011	1959	1939	1901	1927	1813	1869	1934	2077	1949
2,4	1942	1950	1998	1863	1878	1832	1987	1909	1841	1890	2010	1979	1939	1901	1927	1813	1869	1934	2077	1949
2,45	1942	1950	1998	1902	1878	1832	1936	1909	1841	1890	2004	1979	1963	1901	1927	1813	1869	1934	2077	1949
2,5	1942	1950	1960	1902	1878	1832	1868	1909	1841	1896	2004	1979	1962	1901	1880	1813	1869	1934	2077	1970

	Valores de γ																				
Valores de β	1	1,01	1,02	1,03	1,04	1,05	1,06	1,07	1,08	1,09	1,10	1,11	1,12	1,13	1,14	1,15	1,16	1,17	1,18	1,19	1,2
1,7	1784	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785	1785
1,71	1785	1785	1785	1785	1785	1785	1785	1785	1785	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822
1,72	1785	1785	1785	1785	1785	1785	1785	1785	1785	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822
1,73	1785	1785	1785	1785	1785	1785	1785	1785	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822	1822
1,74	1785	1785	1785	1785	1806	1785	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,75	1806	1806	1806	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,76	1806	1806	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,77	1806	1806	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,78	1806	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,79	1806	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,8	1806	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,81	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,82	1806	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,83	1806	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,84	1806	1742	1742	1742	1754	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742	1742
1,85	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754
1,86	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1764	1764	1764	1764	1764	1764	1764	1764
1,87	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1754	1764	1764	1764	1764	1764	1764	1764
1,88	1767	1767	1767	1754	1767	1767	1767	1767	1767	1767	1767	1767	1777	1777	1777	1777	1777	1777	1777	1777	1777

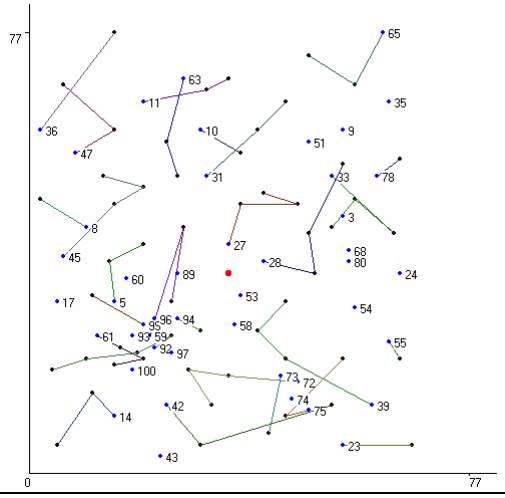
Anexo IV

Soluciones para los problemas *R* y *RC*

R101

Número inicial de rutas = 13
Beta = 1
Gamma = 1

Número final de rutas = 46
Distancia total = 2805,94



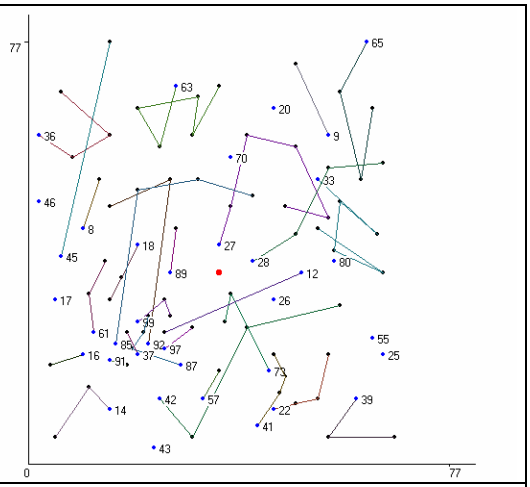
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	103,56	129,52	139,36	142,24	120,18	177,5	202,8	169,21	153,18	208,7	143,41	112,81	161,93	186,23	141,07	167,62	185	143,86	184	195,61	137,06	137,63	124,93
Distancia	71,89	109,32	79,43	90,92	62,06	110,16	82,8	87,96	76,29	74,95	71,99	50,21	98,24	43,84	63,62	68,2	83,19	91,92	71,05	65,15	59,18	52,12	53,85
Carga	38	14	86	54	40	60	73	19	68	57	65	49	74	45	74	57	38	54	55	63	14	16	10
Espera	1,67	0,2	9,92	21,33	28,12	37,34	80,01	51,24	36,89	93,75	31,42	32,61	33,69	102,4	37,46	69,42	71,81	31,94	82,95	90,46	57,88	65,51	61,07
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	63	36	92	14	59	65	45	42	39	33	72	5	47	27	28	31	11	23	75	61	73	99	51
Nodo 3	62	64	95	44	52	71	82	15	21	29	2	83	19	69	12	30	90	67	22	85	41	84	0
Nodo 4	88	0	98	38	6	66	7	56	40	79	87	18	49	50	76	20	32	0	4	37	0	0	
Nodo 5	0		16	0	0	0	48	0	26	77	57	0	0	1	81	0	0		0	91			
Nodo 6			86				0		0	0	0			0	0					0			
Nodo 7			0																				

Nombre	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38	R39	R40	R41	R42	R43	R44	R45	R46
Tiempo	162,47	109,47	163,06	139,02	180,18	148,36	213,1	176,18	160,2	160,26	215,54	172,8	173,38	194,04	183,6	193	197,41	190,03	195	213,1	219,04	218,25	219,06
Distancia	71,15	8,94	72,12	64,03	27,69	44,72	54,65	68,35	34,41	30,53	67,61	45,61	42,76	82,07	49,19	60	60,83	36,06	18	42,19	48,08	40,5	18,11
Carga	10	14	17	16	50	13	21	7	12	11	8	18	36	8	8	3	2	3	15	6	17	22	18
Espera	71,32	90,53	70,94	64,98	132,49	93,64	138,44	97,82	115,8	119,74	127,93	117,2	120,62	101,96	124,4	123	126,59	143,97	167	160,9	160,96	167,75	190,94
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	8	53	78	9	94	3	10	43	97	96	55	54	68	35	74	24	17	60	89	80	100	93	58
Nodo 3	46	0	34	0	13	0	70	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 4	0		0		0		0				0												
Nodo 5																							

R102

Número inicial de rutas = 8
Beta = 1
Gamma = 1

Número final de rutas = 37
Distancia total = 2549,17



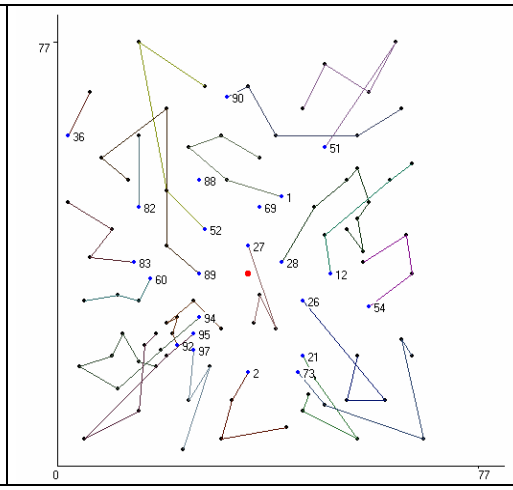
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	188,3	161,93	129,41	142,24	194,04	138,54	172,8	143,86	166,38	215,97	162,52	139,36	148,8	219,06	134,61	207,09	164,52	152,29	177,5	184	204,66	144,83	162,47
Distancia	116,94	112,65	84,06	90,92	130,91	115,7	97,34	98,44	74,38	101,87	75,28	70,83	60,37	49,93	39,1	47,61	49,9	63,54	86,83	68,42	61,74	47,03	70,94
Carga	83	79	27	54	46	25	40	85	79	96	69	54	31	41	39	41	75	45	41	61	49	14	1
Espera	11,36	9,28	15,35	21,33	23,13	2,85	35,46	15,43	32	54,1	47,24	48,52	58,43	139,12	65,51	129,48	74,62	68,75	70,67	75,58	102,93	77,81	81,53
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	63	36	92	14	65	45	42	39	27	33	28	16	61	73	99	18	87	8	9	22	41	57	46
Nodo 3	62	47	88	44	71	64	15	23	69	29	76	86	84	53	6	60	98	48	66	75	74	2	0
Nodo 4	11	19	82	38	78	0	40	67	30	79	81	0	83	58	94	5	59	0	0	56	72	0	
Nodo 5	90	49	0	0	35		54	0	51	68	34		0	0	0	0	96			4	21		
Nodo 6	10	0			0		0		3	24	0						0			0	0		
Nodo 7	32								50	77													
Nodo 8	0								0	0													

Nombre	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37
Tiempo	167,62	176,18	180,18	176,46	197,41	215,54	215,38	213,1	213,1	219,04	128,16	218,25	77,18	32,36
Distancia	63,25	68,35	34,79	60,93	60,83	67,08	28,38	42,19	42,19	52,82	88,16	45,93	57,18	22,36
Carga	9	7	35	2	2	6	24	5	6	18	83	30	39	17
Espera	94,38	97,82	125,39	105,54	126,59	138,46	167	160,9	160,9	146,23	0	152,31	0	0
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	20	43	97	55	17	25	89	70	80	91	85	37	12	26
Nodo 3	0	0	13	0	0	0	52	0	0	100	7	93	95	0
Nodo 4			0				0			0	31	0	0	
Nodo 5											1			
Nodo 6											0			

R103

Número inicial de rutas = 20
Beta = 1,74
Gamma = 1,06

Número final de rutas = 23
Distancia total = 1742,53

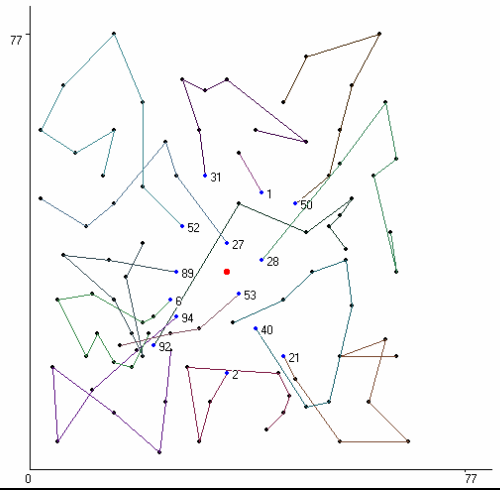


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	161,93	180,18	187,57	159,9	162,99	184	208,7	142,35	135,86	183,6	219,06	213,1	146,21	219,04	72,17	215,54	218,25	194,04	103,24	159,68	197,41	88,34	213,1
Distancia	94,28	50,67	110,24	80,46	83,46	74,86	69,34	98,02	72,14	75,04	41,67	70,37	91,02	83,79	24,33	102,49	88,78	112,49	38,47	78,75	63,12	68,34	70,4
Carga	35	76	79	37	50	73	152	33	27	91	57	77	102	140	6	60	127	71	9	49	38	33	36
Espera	47,66	69,51	27,32	39,44	39,53	69,14	59,36	4,33	23,72	58,56	137,38	92,72	5,19	55,26	37,83	63,05	69,47	31,55	54,76	40,93	94,29	0	102,69
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	36	92	51	83	97	26	28	52	2	21	27	1	89	94	69	73	95	90	88	12	60	82	54
Nodo 3	49	59	65	45	42	39	50	7	57	72	40	31	18	98	0	75	37	32	0	76	5	19	24
Nodo 4	0	96	71	8	87	56	33	64	15	23	53	62	11	44		67	38	30		78	84	0	29
Nodo 5		99	66	46	43	4	81	63	41	22	58	10	47	86		55	14	9		34	17		80
Nodo 6		6	20	0	0	0	79	0	0	74	0	70	48	16		25	85	35		0	0		0
Nodo 7		13	0				3			0		0	0	61		0	93	0					
Nodo 8		0					68							91			0						
Nodo 9							77							100									
Nodo 10							0							0									

R104

Número inicial de rutas = 5
Beta = 1,16
Gamma = 1

Número final de rutas = 14
Distancia total = 1326,62

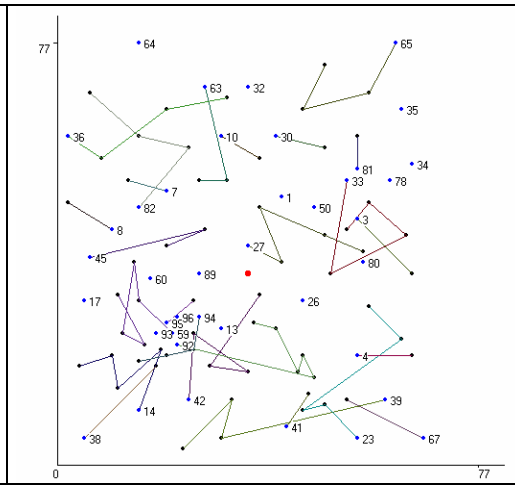


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
Tiempo	225,38	215,27	218,09	163,52	222,99	222,4	229,51	226,34	219,71	181,51	166,48	169,29	219,06	64,39
Distancia	103,35	101,03	121,49	91,84	122,21	115,29	83,2	51,03	105,74	111,51	96,48	89,29	89,77	44,39
Carga	107	88	148	70	150	150	110	98	109	79	110	113	111	15
Espera	52,03	34,23	16,6	11,68	10,79	17,11	46,32	135,31	43,97	0	0	0	49,28	0
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	2	21	27	52	94	6	53	50	28	31	89	40	1
Nodo 3	69	57	72	88	7	98	96	13	33	81	10	83	75	70
Nodo 4	76	15	23	62	11	44	99	95	9	35	63	45	56	0
Nodo 5	79	87	67	82	64	38	84	85	71	34	90	5	54	
Nodo 6	3	73	39	8	49	86	17	0	65	78	32	93	80	
Nodo 7	77	74	55	46	36	14	16		66	24	51	37	12	
Nodo 8	68	22	4	0	47	43	61		20	29	30	60	26	
Nodo 9	0	41	25		19	42	91		0	0	0	18	58	
Nodo 10		0	0		48	97	100					0	0	
Nodo 11					0	0	59							
Nodo 12							0							

R105

Número inicial de rutas = 8
Beta = 1
Gamma = 1

Número final de rutas = 40
Distancia total = 2682,07



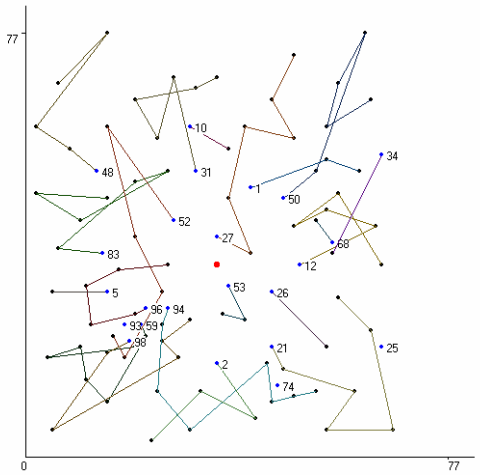
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	185,44	106,63	145,61	127,1	147,04	199,06	104,17	152,11	163,38	198,7	166,18	164,41	177,29	117,15	119,52	192,8	199,04	159,21	110,58	142,02	152,47	127,06	173,6
Distancia	122,04	76,63	85,61	77,1	107,04	73,85	74,17	102,11	64,47	92,99	118,96	102,73	102,16	61,64	93,04	56,3	84,85	88,72	35,08	65,6	71,15	62,13	60,67
Carga	69	46	126	72	47	74	37	102	87	76	53	82	85	31	9	41	33	31	12	42	10	3	13
Espera	23,41	0	0	0	0	65,2	0	0	48,91	55,7	7,22	21,68	25,13	35,5	16,48	116,51	94,19	50,48	55,51	56,41	61,32	54,94	92,93
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	65	63	59	42	36	92	45	14	27	33	39	82	23	30	64	7	38	67	99	81	8	78	41
Nodo 3	71	31	5	95	47	72	52	98	28	12	15	62	75	51	0	48	100	56	6	9	46	0	74
Nodo 4	20	88	83	2	11	21	18	44	69	29	57	19	22	0		0	0	0	0	0	0		0
Nodo 5	66	0	61	87	90	73	0	16	76	79	43	49	55										
Nodo 6	0		85	53	0	40		86	68	77	0	0	54										
Nodo 7			84	0		58		0	0	0			0										
Nodo 8			0			0																	

Nombre	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38	R39	R40
Tiempo	188,81	183	153,06	199,1	140,97	143,18	150,26	175	184,04	202,54	187,41	170,18	176,23	180,03	185	199,1	199,25
Distancia	53,93	66,5	72,11	54,65	33,94	22,36	30,53	68	82,07	68,54	60,83	22,36	30,46	36,06	18	42,19	40,5
Carga	48	16	14	21	13	17	11	23	8	25	2	23	10	3	15	6	22
Espera	94,88	96,5	70,94	124,44	97,03	110,82	109,74	97	91,96	114	116,59	137,82	135,77	133,97	157	146,9	148,75
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	94	3	34	10	50	26	96	32	35	4	17	13	1	60	89	80	93
Nodo 3	97	24	0	70	0	0	0	0	0	25	0	0	0	0	0	0	0
Nodo 4	37	0		0						0							
Nodo 5	91																
Nodo 6	0																

R106

Número inicial de rutas = 4
Beta = 1,1
Gamma = 1

Número final de rutas = 23
Distancia total = 1842,12

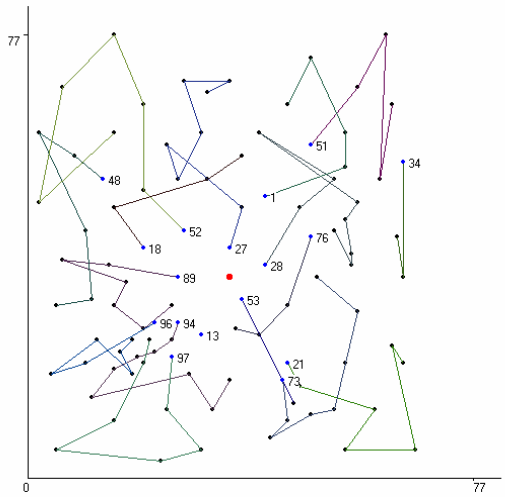


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	184,04	175	179,43	169,48	159,33	194,54	188,07	177,29	183	212,49	190,06	170,18	199,06	141,39	175,08	199,1	198,7	174	173,6	187,41	199,1	199,54	199,25
Distancia	120,36	104,04	99,43	119,48	89,33	102,54	118,07	106,43	102,11	105,3	57,58	91,18	24,36	67,49	88,29	54,65	46,08	50,32	49,19	61,03	77,28	67,08	40,5
Carga	83	94	111	107	130	103	67	141	80	99	58	88	41	39	26	21	50	36	8	28	20	6	22
Espera	3,67	10,96	0	0	0	22	0	0,86	20,89	47,19	72,47	19	144,7	43,9	46,79	124,44	132,62	103,68	114,4	106,38	101,82	122,46	148,75
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	50	31	94	48	59	27	83	21	12	52	96	98	53	1	2	10	68	26	74	5	34	25	93
Nodo 3	33	63	95	47	92	28	45	72	29	19	99	91	40	81	41	70	77	4	0	17	80	0	0
Nodo 4	65	62	42	36	14	69	7	39	3	18	61	38	58	78	57	0	0	0		0	0		
Nodo 5	71	11	15	64	44	30	88	23	76	6	84	87	0	0	43								
Nodo 6	9	90	73	49	16	51	8	67	79	100	60	97			0								
Nodo 7	35	32	22	0	86	20	46	55	24	85	89	13											
Nodo 8	0	0	75		37	66	82	54	0	0	0	0											
Nodo 9			56		0	0	0	0															
Nodo 10			0																				

R107

Número inicial de rutas = 1
Beta = 1,1
Gamma = 1,12

Número final de rutas = 17
Distancia total = 1477,31

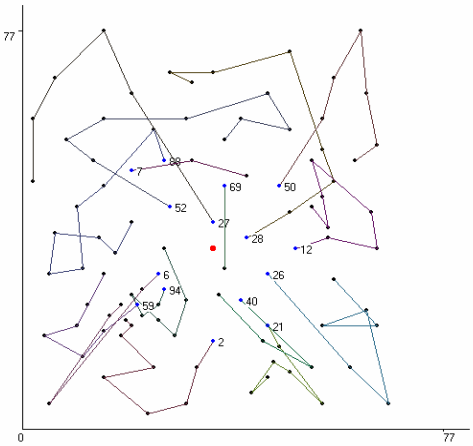


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17
Tiempo	184,04	171,35	185,16	208,21	214,41	139,39	199,54	187,41	202,83	199,06	173,6	177,57	215,11	170,18	122,96	222,55	199,1
Distancia	129,24	81,35	105,16	98,79	99,96	69,39	112,29	108,74	132,04	48,55	49,19	90,98	85,31	22,36	92,96	81,52	69,48
Carga	56	118	104	102	153	83	129	86	83	57	22	86	112	23	26	158	60
Espera	4,8	0	0	29,42	24,44	0	17,25	18,67	0,79	110,5	104,4	36,58	49,8	137,82	0	71,03	89,61
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	51	94	97	27	28	89	21	48	52	76	53	1	73	13	34	96	18
Nodo 3	71	95	42	69	50	83	72	47	7	26	74	81	22	0	24	16	82
Nodo 4	65	92	15	62	33	45	39	36	11	40	0	9	41		29	86	31
Nodo 5	78	98	43	88	30	60	23	8	64	58		66	75		0	61	70
Nodo 6	35	91	38	10	79	5	67	84	49	0		20	56			100	0
Nodo 7	0	44	14	63	3	99	55	17	46			0	4			85	
Nodo 8		87	37	32	68	6	25	0	19				54			93	
Nodo 9		57	59	90	80	0	0		0				12			0	
Nodo 10		2	0	0	77								0				
Nodo 11		0			0												

R108

Número inicial de rutas = 11
Beta = 1,02
Gamma = 1,01

Número final de rutas = 15
Distancia total = 1324,22

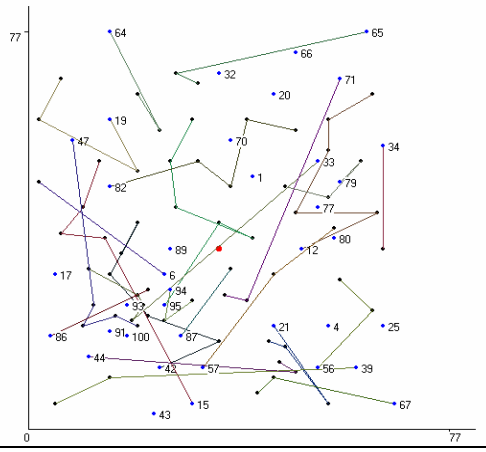


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15
Tiempo	188,77	220,5	207,38	219,63	189,14	155,18	99,47	219,39	146,33	228,53	223,85	206,61	198,7	199,06	88,07
Distancia	61,65	92,23	86,57	107,19	116,3	78,2	32,64	116,34	86,55	111,29	109,08	116,61	90,54	60,96	58,07
Carga	134	89	114	104	73	140	20	118	93	139	89	150	111	42	42
Espera	47,12	38,27	50,81	12,44	12,85	6,98	46,83	33,05	9,77	37,24	44,77	0	28,16	98,1	0
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	94	2	21	88	27	59	69	26	6	28	50	52	12	40	7
Nodo 3	95	57	72	62	11	91	53	39	96	76	9	48	80	56	31
Nodo 4	92	15	23	82	64	44	0	67	38	79	71	47	24	73	1
Nodo 5	99	43	75	8	49	86		55	85	81	65	19	29	58	0
Nodo 6	97	14	74	84	36	16		4	93	66	35	10	33	0	
Nodo 7	87	42	41	17	46	61		25	0	32	34	20	3		
Nodo 8	13	100	22	45	0	5		54		63	78	51	68		
Nodo 9	89	37	0	83		0		0		90	0	30	77		
Nodo 10	0	98		60						0		70	0		
Nodo 11		0		18								0			
Nodo 12				0											

R109

Número inicial de rutas = 8
Beta = 1
Gamma = 1

Número final de rutas = 38
Distancia total = 2892,58



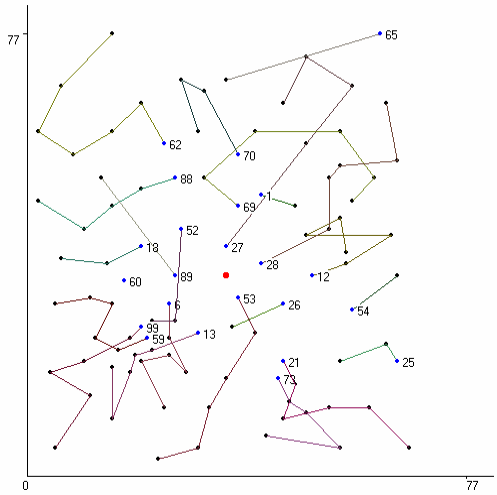
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22
Tiempo	140,63	152,55	159,17	162,03	172,53	131,92	153,06	161,24	198,54	157,6	152,94	176,65	166,06	162,62	144,67	119,28	152,09	139,26	152,58	160,67	167,59	166
Distancia	109,15	122,55	99,17	98,05	117,95	72,94	97,38	111,24	115,54	98,21	102,94	116,66	99,08	132,62	99,53	38,67	65,65	70,74	35,71	75,85	80,99	86,06
Carga	40	33	86	55	80	74	80	65	91	44	108	57	42	67	48	40	52	46	62	4	60	17
Espera	1,48	0	0	3,97	4,59	18,97	5,68	0	23	29,39	0	19,98	36,98	0	15,14	60,62	46,45	48,52	86,88	64,82	56,6	59,94
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	64	65	95	42	15	21	82	33	12	44	47	19	71	39	67	87	79	86	94	6	57	34
Nodo 3	62	63	27	2	83	23	31	98	29	75	61	7	40	14	22	53	78	96	97	46	26	24
Nodo 4	11	90	28	92	45	72	69	59	76	74	16	36	58	38	41	0	3	0	13	0	68	0
Nodo 5	0	0	52	5	8	73	30	99	81	0	85	49	0	0	0		50		0		0	
Nodo 6			88	18	48	0	51	84	9		37	0					0					
Nodo 7			10	60	0		0	0	35		0											
Nodo 8			0	0					0													

Nombre	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38
Tiempo	146,62	154,5	155,18	182,15	160	158	164,54	164,41	167,61	157,04	158,23	154,1	166,1	159,25	165,7	158
Distancia	63,25	80,99	68,35	75,36	68	50	67,08	60,83	51,22	48,08	30,46	42,19	42,19	40,5	39,4	18
Carga	9	25	7	26	23	19	6	2	1	17	10	5	6	22	14	15
Espera	73,38	63,5	76,82	76,79	82	98	87,46	93,59	106,39	98,96	117,77	101,9	113,9	108,75	116,3	130
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	20	66	43	56	32	4	25	17	91	100	1	70	80	93	77	89
Nodo 3	0	0	0	55	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 4				54												
Nodo 5				0												

R110

Número inicial de rutas = 16
Beta = 1,49
Gamma = 1

Número final de rutas = 23
Distancia total = 1706,77

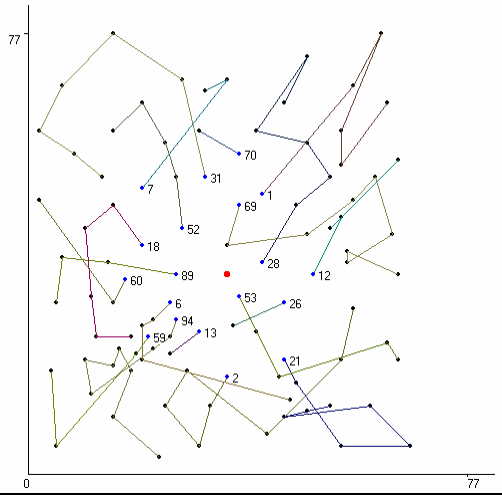


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	168,81	148,23	163,14	191,54	177,6	192,71	188,03	188,11	168,58	177,4	150,93	194,09	175,8	144,26	145,45	176,81	173,3	155,9	181	160	123,54	197,2	158,03
Distancia	90,61	88,23	82,7	67,67	79,93	75,18	69,21	92,85	78,7	96,9	59,74	124,09	74,99	46,61	73,08	97,83	38,53	30,08	62,8	112,09	70,69	58,2	36,06
Carga	75	96	40	74	96	52	73	89	69	119	39	119	117	47	34	116	23	35	21	43	27	51	3
Espera	28,2	0	30,44	63,87	37,68	57,53	58,82	35,26	39,88	20,51	61,19	0	40,81	67,66	32,38	18,97	114,77	105,82	98,2	27,91	22,85	119	111,97
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	27	69	88	6	12	53	13	28	73	99	18	62	59	52	70	21	1	26	54	65	25	89	60
Nodo 3	51	31	7	95	80	40	92	77	74	93	83	11	85	94	90	72	50	58	24	32	55	48	0
Nodo 4	71	30	82	87	29	2	98	33	75	16	45	19	61	96	63	22	0	0	0	0	4	0	
Nodo 5	66	9	8	97	76	57	100	81	23	86	0	47	5	0	10	56					0		
Nodo 6	20	78	46	37	3	15	14	34	41	44		36	84		0	39							
Nodo 7	0	79	0	42	68	43	91	35	0	38		49	17			67							
Nodo 8		0		0	0	0	0	0		0		64	0			0							
Nodo 9												0											

R111

Número inicial de rutas = 3
Beta = 1,34
Gamma = 1,16

Número final de rutas = 19
Distancia total = 1602,31

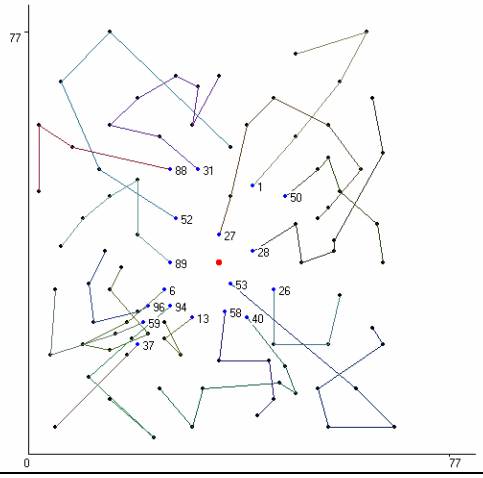


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19
Tiempo	242,35	209,92	192,48	227,8	143,47	181,62	166,7	142,39	190	183,36	172,36	209,72	189,39	176,56	123,72	190,61	162,14	197,7	197,06
Distancia	104,58	115,75	110,59	110,97	73,84	98,61	67,7	91,06	75,18	112,44	76,18	77,2	129,39	54,65	82,14	34,79	80,17	76,99	30,08
Carga	172	95	163	115	66	105	44	89	39	144	79	40	95	21	31	35	30	60	35
Espera	37,77	14,17	1,9	26,83	19,63	13,01	59	11,33	64,82	0,92	36,18	82,53	0	101,9	11,58	135,82	51,97	80,71	146,97
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	94	2	21	69	52	28	89	59	6	31	18	53	1	70	7	13	60	12	26
Nodo 3	95	57	72	27	88	50	83	98	96	63	82	40	71	10	32	97	5	3	58
Nodo 4	92	15	23	76	62	33	45	38	99	64	8	73	65	0	90	0	46	34	0
Nodo 5	44	42	67	79	11	51	17	86	37	49	84	55	9		0		0	77	
Nodo 6	16	87	39	78	19	30	0	0	74	36	61	25	81					0	
Nodo 7	91	41	22	29	0	66			0	47	93	0	35						
Nodo 8	85	4	75	80		20				48	0		0						
Nodo 9	100	54	56	68		0				0									
Nodo 10	14	0	0	24															
Nodo 11	43			0															
Nodo 12	0																		

R112

Número inicial de rutas = 18
Beta = 1,4
Gamma = 1

Número final de rutas = 18
Distancia total = 1517,79

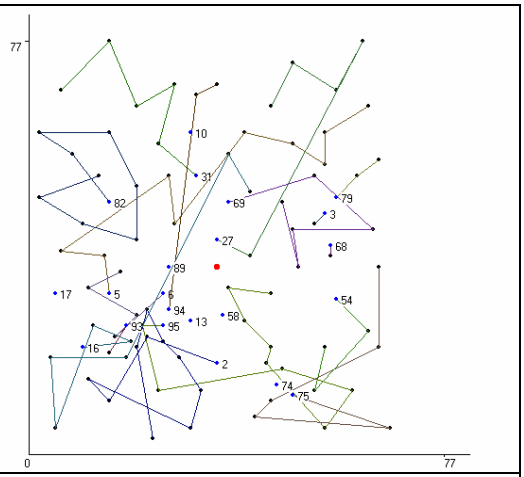


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18
Tiempo	181,93	184,36	186,7	212,24	183,01	180,64	186,77	228,5	184,59	198,25	129,1	186,42	193,6	186,88	192,23	151,22	134,12	179,02
Distancia	85,62	60,62	109,66	104,47	75,19	84,85	90,82	111,98	100,75	80,22	78,29	87,6	74,6	106,88	67,29	43,4	92,35	63,2
Carga	98	51	89	80	76	41	82	113	112	85	106	80	73	127	57	81	42	65
Espera	16,31	73,74	27,04	57,77	37,82	65,79	45,96	46,53	13,85	58,03	10,81	28,82	59	0	74,94	67,82	1,76	75,82
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	27	96	52	1	6	37	94	53	28	50	59	40	89	31	58	13	88	26
Nodo 3	69	99	48	51	93	100	98	39	76	33	85	72	18	62	2	97	47	21
Nodo 4	30	61	49	71	16	38	44	67	12	81	86	75	7	19	73	87	36	4
Nodo 5	20	84	64	65	91	0	43	23	80	79	17	74	82	11	22	95	46	54
Nodo 6	9	83	70	66	92		14	56	68	29	0	57	8	63	41	0	0	0
Nodo 7	78	0	0	0	5		0	25	34	24		15	45	90	0			
Nodo 8	3				60			55	35	0		42	0	10				
Nodo 9	77				0			0	0			0		32				
Nodo 10	0													0				

R201

Número inicial de rutas = 7
Beta = 1,8
Gamma = 1

Número final de rutas = 22
Distancia total = 1882,40



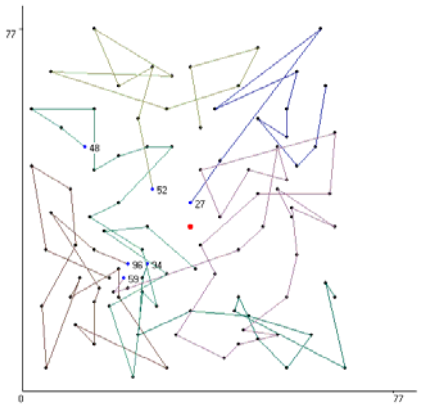
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22
Tiempo	652,57	609,18	554,93	737,04	717,43	810,8	573,97	893,78	882	749,03	575,06	704	846,7	612,5	737,6	870,1	715,6	711,18	862,44	773,41	774	868,06
Distancia	117,18	129,25	115,81	163,52	156,66	135,5	95,21	158,89	143,44	63,91	72,27	90,83	44,89	50,99	79,4	44,47	49,19	22,36	51,69	60,83	18	18,11
Carga	101	198	107	152	126	128	71	125	75	63	40	53	27	16	45	42	8	23	23	2	15	18
Espera	475,38	369,93	379,12	473,52	450,76	585,3	418,76	654,89	678,56	635,12	472,79	583,17	781,81	551,5	618,2	805,62	656,4	678,82	790,75	702,59	746	839,94
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	27	95	31	5	2	82	69	16	75	6	79	94	3	10	54	68	74	13	93	17	89	58
Nodo 3	28	59	62	83	92	47	33	98	67	85	78	90	77	0	55	80	0	0	91	0	0	0
Nodo 4	65	42	63	45	14	36	29	61	41	99	34	32	0		56	0			0			
Nodo 5	71	72	11	88	44	19	76	38	22	84	0	0			4							
Nodo 6	66	39	64	52	15	7	12	86	25	60					0							
Nodo 7	20	23	49	30	57	18	50	100	24	0												
Nodo 8	0	73	0	51	87	8	0	70	0													
Nodo 9		21		81	97	46		1														
Nodo 10		40		9	96	48		0														
Nodo 11		53		35	37	0																
Nodo 12		26		0	43																	
Nodo 13		0			0																	

R202

Número inicial de rutas = 6
Beta = 1,85
Gamma = 1,05

Número final de rutas = 6
Distancia total = 1377,59

Nombre	R1	R2	R3	R4	R5	R6
Tiempo	737,04	860,54	819,10	841,25	870,10	909,27
Distancia	191,52	176,60	201,82	255,14	320,11	232,40
Carga	168	215	170	282	334	289
Espera	435,52	553,94	497,27	396,11	289,98	486,87
Nodo 1	0	0	0	0	0	0
Nodo 2	27	94	52	96	59	48
Nodo 3	1	95	62	5	98	47
Nodo 4	65	42	63	45	92	36
Nodo 5	30	2	11	85	26	19
Nodo 6	71	72	64	91	12	82
Nodo 7	9	39	90	14	33	7
Nodo 8	81	23	49	44	3	88
Nodo 9	51	75	10	15	50	31
Nodo 10	79	21	20	37	28	18
Nodo 11	78	73	66	99	69	83
Nodo 12	35	67	32	16	34	6
Nodo 13	0	55	70	61	29	87
Nodo 14		25	0	38	76	97
Nodo 15		0		86	53	43
Nodo 16				84	40	100
Nodo 17				8	57	13
Nodo 18				46	41	60
Nodo 19				17	22	89
Nodo 20				93	56	58
Nodo 21				0	74	0
Nodo 22					4	
Nodo 23					54	
Nodo 24					68	
Nodo 25					24	
Nodo 26					77	
Nodo 27					80	
Nodo 28					0	

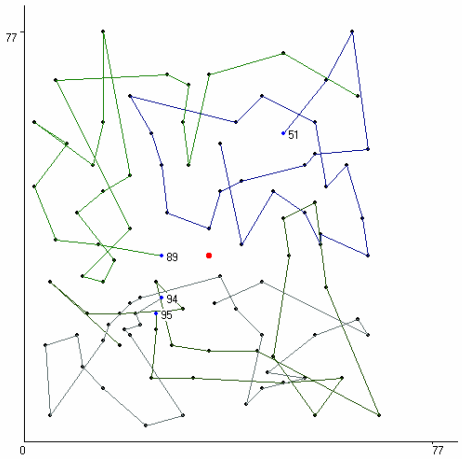


R203

Número inicial de rutas = 4
Beta = 2,03
Gamma = 1,16

Número final de rutas = 4
Distancia total = 1160,75

Nombre	R1	R2	R3	R4
Tiempo	955,09	907,95	737,04	888,26
Distancia	293,36	290,79	329,59	247,01
Carga	351	357	357	393
Espera	401,73	397,17	167,44	361,26
Nodo 1	0	0	0	0
Nodo 2	51	95	89	94
Nodo 3	71	97	83	59
Nodo 4	65	42	45	92
Nodo 5	34	57	46	98
Nodo 6	81	75	47	37
Nodo 7	33	39	36	15
Nodo 8	1	23	48	43
Nodo 9	69	72	19	14
Nodo 10	27	12	64	44
Nodo 11	52	76	7	16
Nodo 12	88	3	82	86
Nodo 13	62	54	8	38
Nodo 14	11	67	60	91
Nodo 15	30	73	5	85
Nodo 16	20	2	84	99
Nodo 17	9	87	18	96
Nodo 18	79	6	49	53
Nodo 19	78	13	63	40
Nodo 20	29	93	90	21
Nodo 21	24	61	10	41
Nodo 22	68	17	31	22
Nodo 23	80	100	32	56
Nodo 24	77	0	66	74
Nodo 25	50		35	4
Nodo 26	28		0	55
Nodo 27	70			25
Nodo 28	0			26
Nodo 29				58
Nodo 30				0

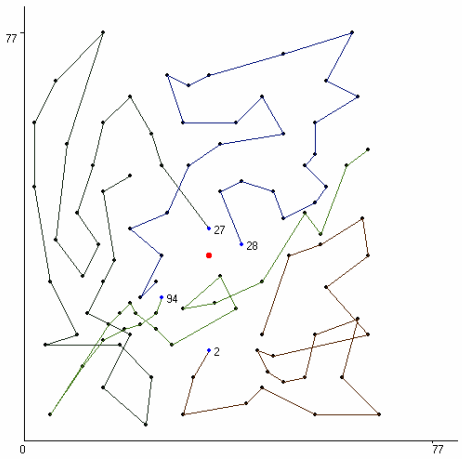


R204

Número inicial de rutas = 4
Beta = 1,47
Gamma = 1

Número final de rutas = 4
Distancia total = 953,67

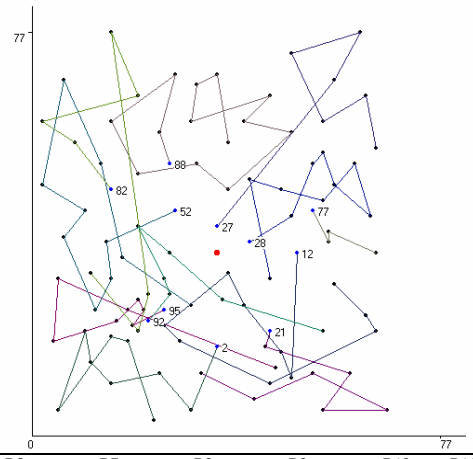
Nombre	R1	R2	R3	R4
Tiempo	994,10	965,71	997,67	789,67
Distancia	202,05	221,30	305,06	225,26
Carga	380	289	400	389
Espera	572,05	524,41	412,62	284,41
Nodo 1	0	0	0	0
Nodo 2	94	2	27	28
Nodo 3	95	57	88	69
Nodo 4	92	15	62	1
Nodo 5	98	41	11	50
Nodo 6	91	22	19	76
Nodo 7	44	23	48	3
Nodo 8	38	67	8	79
Nodo 9	85	39	83	33
Nodo 10	93	55	84	81
Nodo 11	99	4	45	9
Nodo 12	59	56	47	35
Nodo 13	97	75	64	71
Nodo 14	87	74	49	65
Nodo 15	40	73	36	66
Nodo 16	53	72	46	32
Nodo 17	13	25	17	90
Nodo 18	58	54	16	63
Nodo 19	26	24	86	10
Nodo 20	77	29	100	30
Nodo 21	68	80	42	20
Nodo 22	78	12	43	51
Nodo 23	34	21	14	70
Nodo 24	0	0	37	31
Nodo 25			61	52
Nodo 26			5	18
Nodo 27			60	89
Nodo 28			82	96
Nodo 29			7	6
Nodo 30			0	0



R205

Número inicial de rutas = 10
Beta = 1,94
Gamma = 1,36

Número final de rutas = 11
Distancia total = 1471,00



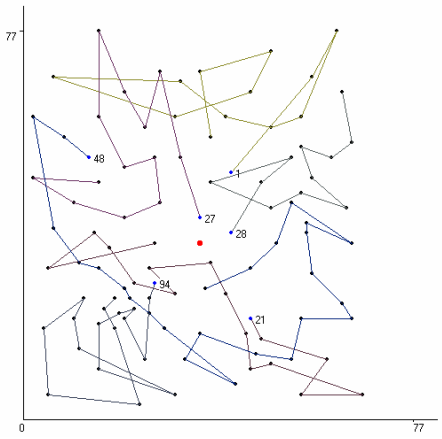
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11
Tiempo	595,46	696,25	759,10	803,68	708,60	795,43	794,99	780,89	604,82	806,38	818,56
Distancia	120,67	134,51	176,30	126,76	109,73	153,77	147,34	158,42	169,93	105,32	68,25
Carga	123	114	180	175	154	177	118	167	95	96	59
Espera	394,79	491,74	442,79	586,92	488,87	531,66	547,64	512,47	354,89	631,06	710,30
Nodo 1	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	21	27	88	95	28	52	2	12	82	92	77
Nodo 3	73	71	62	98	76	83	15	75	47	94	80
Nodo 4	39	65	63	59	33	5	42	72	36	6	68
Nodo 5	23	66	19	99	81	61	14	40	11	18	24
Nodo 6	67	9	7	85	79	45	44	53	64	89	0
Nodo 7	56	35	31	86	29	8	16	97	96	58	
Nodo 8	41	34	69	17	78	46	38	87	37	4	
Nodo 9	57	0	51	93	3	49	91	22	84	0	
Nodo 10	0		30	74	50	48	100	25	0		
Nodo 11			20	0	1	60	43	55			
Nodo 12			10		26	13	0	54			
Nodo 13			90		0	0		0			
Nodo 14			32								
Nodo 15			70								
Nodo 16			0								

R206

Número inicial de rutas = 6
Beta = 1,29
Gamma = 1,14

Número final de rutas = 6
Distancia total = 1163,87

Nombre	R1	R2	R3	R4	R5	R6
Tiempo	762,08 896,94	759,10	508,36	681,04	782,55	
Distancia	184,72 247,17	219,58	174,99	149,86	187,55	
Carga	257 383	203	171	155	289	
Espera	407,36 389,78	409,52	193,37	411,18	415,00	
Nodo 1	0	0	0	0	0	0
Nodo 2	21	1	27	28	94	48
Nodo 3	72	71	31	50	95	47
Nodo 4	39	65	63	33	42	36
Nodo 5	23	9	62	69	37	45
Nodo 6	67	51	11	76	92	84
Nodo 7	75	30	64	3	98	5
Nodo 8	22	90	19	29	91	99
Nodo 9	73	49	7	79	14	59
Nodo 10	40	10	88	81	15	97
Nodo 11	53	20	52	78	44	87
Nodo 12	6	66	18	34	16	41
Nodo 13	13	32	8	35	61	57
Nodo 14	96	70	46	0	86	2
Nodo 15	60	0	82		38	74
Nodo 16	83		0		43	56
Nodo 17	17				100	4
Nodo 18	89				85	25
Nodo 19	0				93	55
Nodo 20					0	54
Nodo 21						80
Nodo 22						68
Nodo 23						24
Nodo 24						77
Nodo 25						12
Nodo 26						26
Nodo 27						58
Nodo 28						0

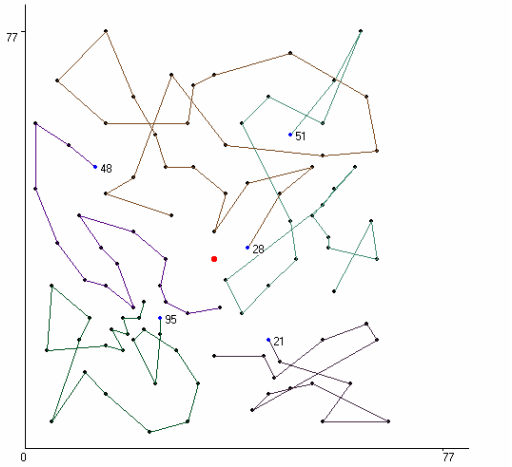


R207

Número inicial de rutas = 5
Beta = 1,25
Gamma = 1,06

Número final de rutas = 5
Distancia total = 1021,43

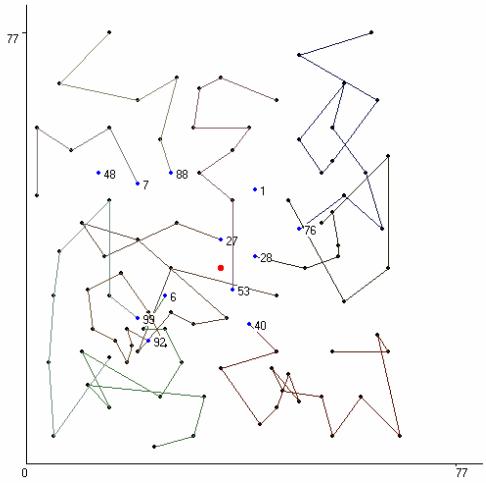
Nombre	R1	R2	R3	R4	R5
Tiempo	831,02	850,86	844,44	830,47	759,06
Distancia	166,29	239,68	283,26	182,76	149,44
Carga	219	288	355	348	248
Espera	514,74	411,18	311,18	417,72	439,62
Nodo 1	0	0	0	0	0
Nodo 2	21	51	28	95	48
Nodo 3	72	71	50	97	47
Nodo 4	39	65	33	42	36
Nodo 5	23	9	1	37	46
Nodo 6	67	20	27	92	45
Nodo 7	56	30	69	87	84
Nodo 8	75	76	31	57	5
Nodo 9	22	12	88	15	99
Nodo 10	41	26	62	43	60
Nodo 11	25	40	11	14	83
Nodo 12	55	53	64	44	8
Nodo 13	4	3	49	38	18
Nodo 14	74	79	19	16	89
Nodo 15	73	78	10	61	6
Nodo 16	2	77	90	17	94
Nodo 17	0	68	32	86	13
Nodo 18		80	66	91	58
Nodo 19		24	35	100	0
Nodo 20		29	34	85	
Nodo 21		54	81	98	
Nodo 22		0	70	93	
Nodo 23			63	59	
Nodo 24			7	96	
Nodo 25			82	0	
Nodo 26			52		
Nodo 27			0		



R209

Número inicial de rutas = 10
Beta = 2,64
Gamma = 1

Número final de rutas = 12
Distancia total = 1346,86



Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9
Tiempo	622,18	699,83	523,25	664,39	710,56	503,47	710,61	648,90	548,66
Distancia	139,10	133,53	118,59	104,43	117,31	95,14	129,58	159,03	179,66
Carga	161	165	89	124	152	55	121	228	179
Espera	373,07	456,30	344,66	469,96	493,25	358,33	501,03	329,87	248,99
Nodo 1	0	0	0	0	0	0	0	0	0
Nodo 2	6	27	88	53	28	7	99	40	76
Nodo 3	59	52	62	69	12	19	5	21	79
Nodo 4	95	83	63	31	80	47	82	2	29
Nodo 5	87	8	11	70	68	36	45	41	78
Nodo 6	42	18	49	30	3	46	17	22	9
Nodo 7	16	58	64	10	77	0	86	72	71
Nodo 8	14	13	0	90	34		38	75	51
Nodo 9	44	94		32	24		91	73	33
Nodo 10	57	37		20	54		0	74	81
Nodo 11	15	89		0	50			56	35
Nodo 12	43	26			0			23	66
Nodo 13	0	0						39	6
Nodo 14								67	0
Nodo 15								55	
Nodo 16								25	
Nodo 17								4	
Nodo 18									

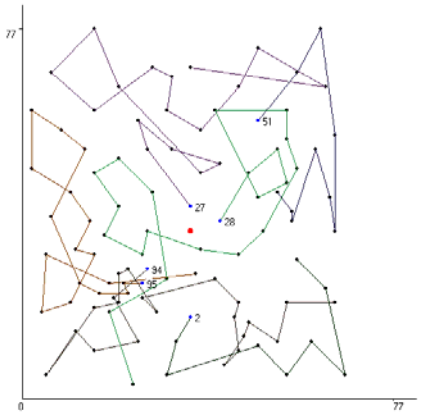
Nombre	R10	R11	R12
Tiempo	713,52	661,23	731,80
Distancia	84,42	30,46	55,61
Carga	138	10	36
Espera	529,09	620,77	666,20
Nodo 1	0	0	0
Nodo 2	92	1	48
Nodo 3	93	0	0
Nodo 4	98		
Nodo 5	100		
Nodo 6	85		
Nodo 7	61		
Nodo 8	84		
Nodo 9	60		
Nodo 10	96		
Nodo 11	97		
Nodo 12	0		

R210

Número inicial de rutas = 6
Beta = 1,46
Gamma = 1

Número final de rutas = 6
Distancia total = 1178,91

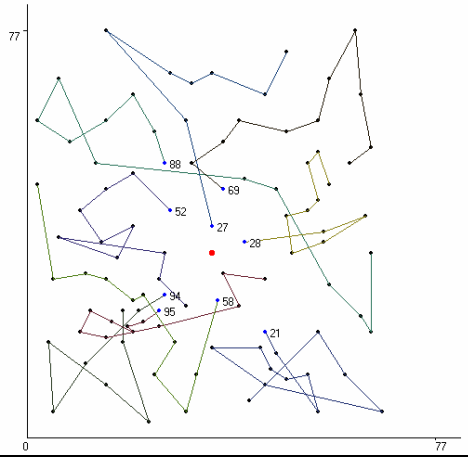
Nombre	R1	R2	R3	R4	R5	R6
Tiempo	873,06	850,70	869,54	671,29	817,39	890,99
Distancia	186,58	152,25	209,29	127,75	246,99	256,05
Carga	338	130	279	145	254	312
Espera	496,48	598,45	440,25	453,54	390,41	414,94
Nodo 1	0	0	0	0	0	0
Nodo 2	95	51	94	2	27	28
Nodo 3	59	71	98	57	88	50
Nodo 4	92	65	42	15	62	33
Nodo 5	85	34	14	75	31	3
Nodo 6	61	24	44	23	1	76
Nodo 7	45	29	38	39	69	30
Nodo 8	48	78	91	67	11	9
Nodo 9	47	80	37	55	64	81
Nodo 10	36	68	99	54	49	79
Nodo 11	46	77	96	0	19	12
Nodo 12	8	0	87		63	26
Nodo 13	83		97		90	53
Nodo 14	84		40		10	89
Nodo 15	5		21		70	6
Nodo 16	16		73		20	60
Nodo 17	86		22		66	18
Nodo 18	17		41		35	82
Nodo 19	93		74		32	7
Nodo 20	58		72		0	52
Nodo 21	0		56			13
Nodo 22			4			100
Nodo 23			25			43
Nodo 24			0			0



R211

Número inicial de rutas = 9
Beta = 1,49
Gamma = 1

Número final de rutas = 9
Distancia total = 1174,14

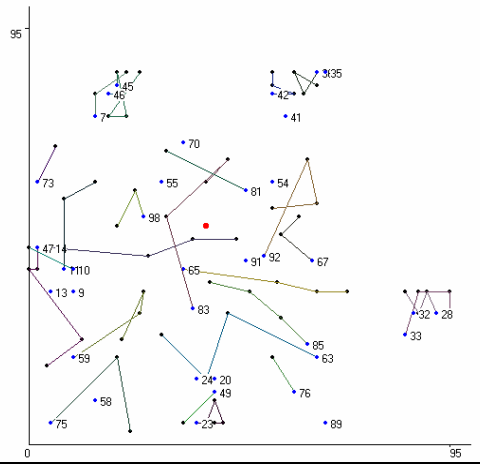


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9
Tiempo	636,17	606,25	676,82	530,06	639,31	555,39	603,19	710,65	554,28
Distancia	194,54	131,87	165,93	122,16	101,77	124,58	93,24	130,70	109,35
Carga	207	111	211	190	186	145	166	120	122
Espera	301,62	394,38	380,89	317,90	427,54	320,81	389,95	469,94	334,93
Nodo 1	0	0	0	0	0	0	0	0	0
Nodo 2	88	27	21	94	28	69	95	58	52
Nodo 3	62	10	72	59	68	31	92	57	7
Nodo 4	11	64	23	44	29	70	98	15	82
Nodo 5	19	63	56	38	80	30	37	42	8
Nodo 6	47	90	75	86	12	51	85	87	83
Nodo 7	36	32	74	14	76	9	61	96	18
Nodo 8	49	20	73	43	77	71	16	99	60
Nodo 9	48	66	2	100	3	65	91	5	45
Nodo 10	1	0	22	93	33	35	97	84	89
Nodo 11	50		67	0	81	34	40	17	6
Nodo 12	54		39		79	78	53	46	13
Nodo 13	55		4		0	0	26	0	0
Nodo 14	25		41				0		
Nodo 15	24		0						
Nodo 16	0								

RC101

Número inicial de rutas = 9
Beta = 1
Gamma = 1

Número final de rutas = 38
Distancia total = 3163,98



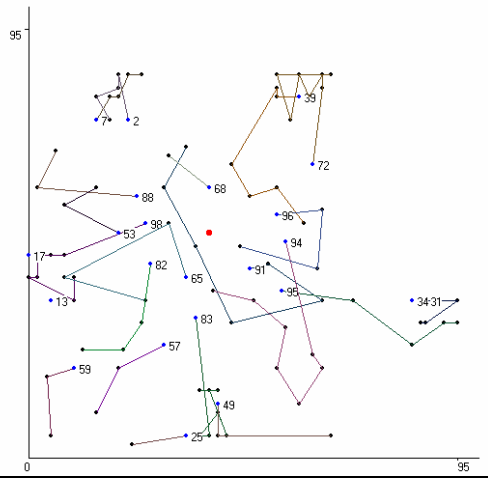
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19
Tiempo	157,52	193,08	176,08	183,52	149,84	209,62	161,34	158,29	174,61	164	180,17	121	191,93	168,43	206,04	209,1	228,24	137,98	144,42
Distancia	74,7	105,22	85,01	132,44	109,84	85,83	83,95	118,29	100,67	77,25	123,49	49,41	84,51	119,75	104,86	61,27	145,46	85,92	86,71
Carga	62	100	50	80	65	38	55	86	110	71	98	33	104	60	110	35	31	50	35
Espera	42,82	37,86	51,07	11,08	0	83,78	37,39	0	33,94	46,75	6,67	41,59	67,41	8,69	51,19	117,83	52,78	32,06	37,71
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	45	42	33	59	14	65	63	36	83	47	98	11	28	23	67	75	76	73
Nodo 3	72	5	39	31	52	82	95	64	38	69	15	88	12	29	21	94	74	51	79
Nodo 4	71	2	44	27	99	90	62	22	40	61	16	53	78	30	19	93	77	0	0
Nodo 5	96	6	43	26	86	80	50	57	37	68	87	0	60	34	18	0	0		
Nodo 6	0	1	0	0	0	0	0	0	0	0	97		0	0	48				
Nodo 7		0									0				0				

Nombre	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38
Tiempo	161,96	209,24	172,26	134,54	132,81	209,28	187,06	199,31	167,06	192,08	193,2	167,03	164,14	190,08	206,48	193,06	209,17	189,04	209,65
Distancia	88,32	51,4	75,43	67,08	61,61	93,23	75,26	83,11	70,11	102,16	88,41	36,06	28,28	76,16	104,96	70,11	94,34	24,08	39,29
Carga	40	13	55	20	20	30	50	50	10	10	20	16	16	10	25	10	20	13	23
Espera	43,64	137,83	56,83	57,46	61,19	96,05	91,8	96,2	86,94	79,92	94,8	120,97	125,86	103,92	91,52	112,94	104,83	154,96	160,35
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	7	81	85	9	41	49	46	10	20	32	35	54	55	13	89	24	58	91	70
Nodo 3	8	100	84	0	0	25	4	17	0	0	0	0	0	0	0	0	0	0	0
Nodo 4	3	0	56			0	0	0											
Nodo 5	0		66																
Nodo 6			0																

RC102

Número inicial de rutas = 8
Beta = 1,24
Gamma = 1

Número final de rutas = 23
Distancia total = 2217,83

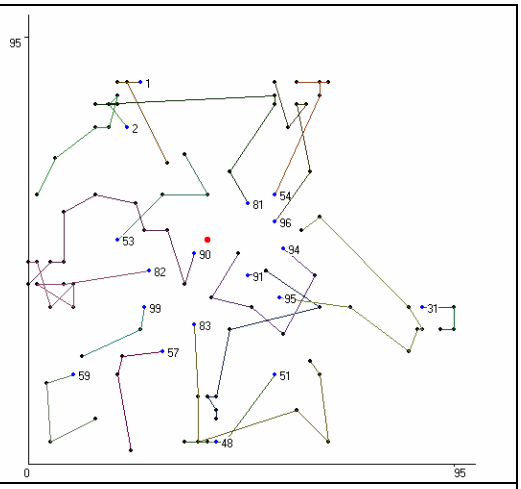


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23
Tiempo	209,65	150,13	209,1	173,27	174,26	136,14	186	215,63	192,08	193,06	209,62	166,51	154,21	191,93	219,35	218,48	217,46	210,05	190,08	209,24	199,31	236,15	104,87
Distancia	114,7	88,52	108,26	123,27	94,26	88,97	106	135,63	117,97	111,02	63,95	87,39	82,34	68,4	96,46	122,53	89,65	118,8	76,16	40,64	80,62	107,42	94,87
Carga	92	90	68	86	139	98	134	178	70	174	62	54	53	49	51	45	100	70	10	24	20	27	30
Espera	24,95	11,61	30,83	0	0	7,17	0	0	24,11	2,03	105,67	49,11	31,87	93,52	92,89	65,95	77,81	61,25	103,92	148,6	108,69	108,72	0
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	91	2	39	95	98	65	94	72	31	83	96	88	82	53	57	49	7	59	13	68	17	25	34
Nodo 3	92	5	42	50	12	69	85	37	29	21	71	73	52	78	74	48	46	97	0	100	0	77	0
Nodo 4	62	45	44	33	14	11	63	36	27	23	67	79	86	60	58	89	4	75		0		0	
Nodo 5	64	8	61	28	47	99	76	38	30	19	80	0	87	0	0	0	3	0					
Nodo 6	90	6	81	26	15	0	51	40	32	18	0		0				1						
Nodo 7	55	0	54	0	16		84	41	0	22							0						
Nodo 8	70		93		9		56	43		20													
Nodo 9	0		0		10		66	35		24													
Nodo 10					0		0	0		0													

RC103

Número inicial de rutas = 9
Beta = 1,23
Gamma = 1,01

Número final de rutas = 18
Distancia total = 1892,09

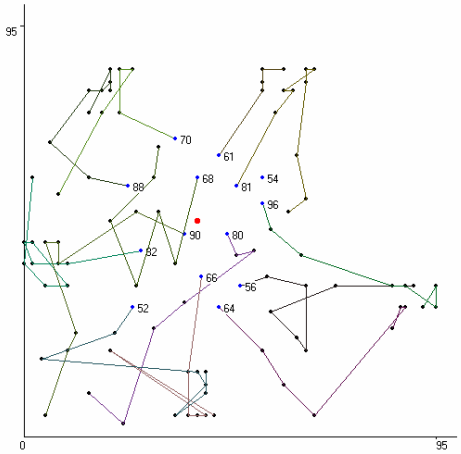


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18
Tiempo	195,57	197,64	187,06	209,1	215,08	193,2	161,35	171	224,75	202,46	209,62	164,63	209,94	228,24	209,17	209,28	112,04	209,24
Distancia	114,16	109,96	121,24	119,74	115,08	101,4	101,35	121	114,26	132,46	78,82	78,15	74,08	107,35	120,14	106,42	92,04	84,44
Carga	147	125	89	113	171	116	114	90	159	118	98	31	65	66	90	59	20	53
Espera	1,41	17,68	5,82	9,36	0	41,81	0	0	30,49	0	70,8	56,48	95,86	80,9	49,03	72,86	0	84,8
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	91	2	81	95	90	54	96	31	82	83	94	99	53	57	59	51	48	1
Nodo 3	92	46	61	50	65	37	72	29	11	24	67	52	55	86	97	18	21	5
Nodo 4	62	45	42	33	69	36	39	27	15	23	84	87	68	74	75	25	0	3
Nodo 5	64	6	44	32	98	40	38	26	9	76	56	0	70	77	58	0		100
Nodo 6	20	7	8	30	88	35	41	28	10	89	66		0	0	0			0
Nodo 7	22	79	4	34	60	0	43	0	13	63	80							
Nodo 8	49	73	0	71	78		0		47	85	0							
Nodo 9	19	0		93	12				17	0								
Nodo 10	0			0	14				0									
Nodo 11					16													
Nodo 12					0													

RC104

Número inicial de rutas = 11
Beta = 2,71
Gamma = 1,74

Número final de rutas = 14
Distancia total = 1696,35

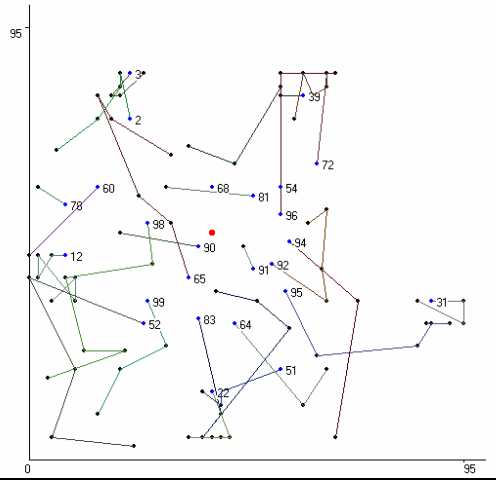


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
Tiempo	227,8	238,46	234,48	227,02	226,43	142,31	209,24	225,41	224,01	230,28	233,36	232,66	173,2	46,06
Distancia	127,23	131,68	149,35	147,02	145,39	84,15	117,59	144,6	106,69	131,35	113,51	148,53	113,2	36,06
Carga	104	144	104	94	133	79	103	122	165	168	176	186	130	16
Espera	30,57	16,78	15,13	0	1,04	8,16	21,65	10,81	17,32	18,93	29,85	4,13	0	0
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	80	82	66	56	90	61	68	64	81	96	88	52	70	54
Nodo 3	91	10	24	95	98	42	65	51	41	94	60	86	2	0
Nodo 4	92	15	23	62	11	44	69	76	38	67	79	97	3	
Nodo 5	83	17	21	63	12	43	99	89	39	34	8	22	1	
Nodo 6	57	47	48	85	14	40	53	30	35	31	46	49	6	
Nodo 7	77	9	74	84	87	0	55	32	36	26	45	25	78	
Nodo 8	58	13	18	50	59		100	33	37	27	4	19	0	
Nodo 9	0	16	0	29	75		0	0	72	28	5	20		
Nodo 10		73		0	0				71	0	7	0		
Nodo 11		0							93		0			
Nodo 12									0					

RC105

Número inicial de rutas = 6
Beta = 1,26
Gamma = 1,01

Número final de rutas = 25
Distancia total = 2259,58



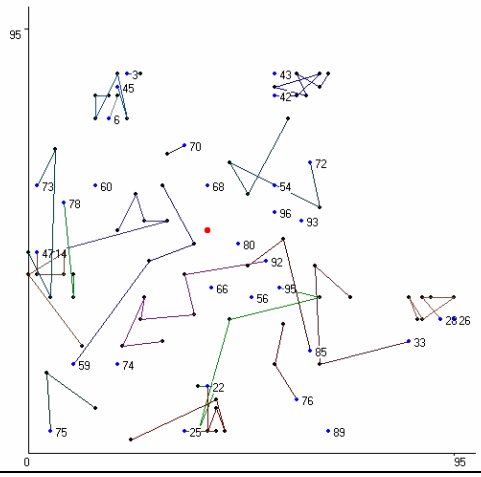
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17
Tiempo	204,1	229,24	201,65	184,27	152,48	161,39	200,08	205,19	154,92	177,09	170,9	229,35	131,26	222,98	156,5	226,28	174,14
Distancia	75,39	97,15	93,2	124,27	102,48	100,55	109,18	134,12	92,53	109,31	120,9	100,82	41,51	158,19	79,06	100,05	45,29
Carga	43	95	62	48	110	138	120	134	54	137	110	66	11	65	32	100	26
Espera	78,7	72,09	58,45	0	0	0,84	20,9	11,07	32,38	7,77	0	88,53	69,76	14,79	57,44	76,23	108,85
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	65	39	95	2	72	12	98	64	83	31	99	90	52	78	22	81
Nodo 3	62	69	42	85	5	36	14	82	76	19	29	57	53	16	73	18	55
Nodo 4	67	88	44	33	45	37	15	11	63	23	27	74	0	59	0	48	0
Nodo 5	71	8	61	30	7	38	47	87	0	84	26	58		75		21	
Nodo 6	93	6	70	28	79	40	9	86		56	34	0		77		25	
Nodo 7	0	100	0	32	0	41	10	97		66	0			0		0	
Nodo 8		0		0		0	13	0		0							
Nodo 9							0										

Nombre	R18	R19	R20	R21	R22	R23	R24	R25
Tiempo	184,06	216,15	204,2	182,74	164	218,52	155,03	207,62
Distancia	90,53	92,91	102,73	119,86	20	88,45	36,06	25,04
Carga	49	80	66	78	21	37	16	26
Espera	53,53	83,24	71,48	32,88	134	110,07	108,97	162,57
Nodo 1	0	0	0	0	0	0	0	0
Nodo 2	51	3	96	94	68	60	54	91
Nodo 3	20	46	43	50	0	17	0	80
Nodo 4	49	4	35	89		0		0
Nodo 5	24	1	0	0				
Nodo 6	0	0						

RC106

Número inicial de rutas = 9
Beta = 1
Gamma = 1

Número final de rutas = 32
Distancia total = 2787,45



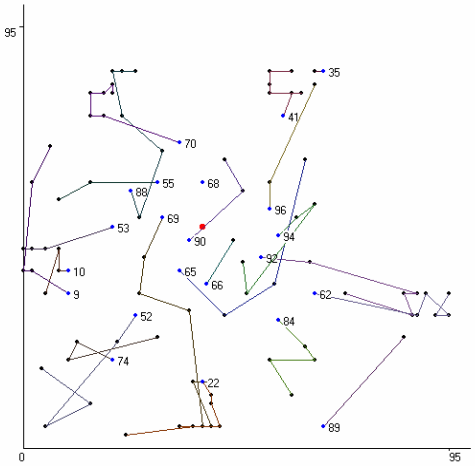
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18
Tiempo	177,44	171,6	210,01	186,24	202,36	180,09	170,46	150,7	152,55	181,97	236,05	185,77	135,67	198,49	174,04	162,72	172,06	178,08
Distancia	107,44	111,6	140,01	126,24	122,36	140,09	118,12	110,7	102,55	121,97	134,05	130,82	88,61	132,74	83,22	90,28	74,74	80,44
Carga	113	100	110	76	160	55	61	50	70	123	127	71	68	65	38	77	60	30
Espera	0	0	0	0	0	0	2,34	0	0	0	42	24,95	17,06	25,75	60,82	42,44	77,31	77,64
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	45	28	95	42	33	72	59	14	47	22	75	76	73	85	78	6	3
Nodo 3	65	2	31	62	39	63	71	82	69	15	21	97	51	79	94	9	4	1
Nodo 4	83	5	29	64	36	67	61	90	98	11	19	58	84	13	91	10	0	0
Nodo 5	52	7	27	23	44	50	81	55	88	12	18	0	0	17	0	0		
Nodo 6	99	8	30	20	38	0	41	0	53	16	49			0				
Nodo 7	86	46	34	24	40		0		0	87	77							
Nodo 8	57	0	32	0	37					0	0							
Nodo 9	0		0		35													
Nodo 10					0													

Nombre	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32
Tiempo	168,52	161,08	143,03	179,48	142,52	141,04	196,04	152,03	172,06	149	176,93	179,1	193,71	179,62
Distancia	117,05	76,16	36,06	104,96	31,05	26,08	97,32	36,06	72,11	20	53,85	42,19	43,35	15,23
Carga	30	20	19	25	26	16	30	16	8	21	17	3	26	13
Espera	41,48	74,92	96,97	64,52	101,48	104,96	78,72	105,97	89,94	119	113,07	126,9	130,35	154,38
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	26	43	56	89	96	66	25	54	74	68	60	93	70	80
Nodo 3	0	0	0	0	0	0	48	0	0	0	0	0	100	0
Nodo 4							0						0	

RC107

Número inicial de rutas = 4
Beta = 2
Gamma = 1

Número final de rutas = 22
Distancia total = 2020,36

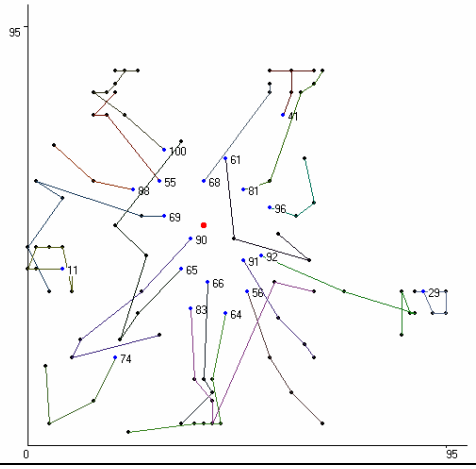


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22
Tiempo	187,15	200,84	214,79	134,68	199,83	181,45	144,15	190,77	185,52	186,47	158,12	179,95	223,5	199,91	172,96	182,7	197,62	221,77	203,39	126,01	131,28	120
Distancia	110,35	130,99	130	94,68	116,84	105,34	44,4	71,89	87,14	101,49	81,12	112,49	123,1	89,56	97,01	95,5	69,76	86,3	130,86	89,22	32,32	20
Carga	113	83	102	47	105	111	25	72	100	76	45	95	167	133	63	130	60	52	35	60	29	21
Espera	16,8	9,85	34,79	0	2,99	6,11	69,76	68,88	58,38	34,98	37	17,46	30,39	40,35	35,94	17,19	97,86	105,48	52,52	16,8	78,96	90
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	62	52	65	69	88	90	94	10	84	53	9	22	70	74	41	55	96	89	35	66	68
Nodo 3	67	30	86	64	82	98	81	93	11	85	14	15	49	6	87	39	60	54	33	36	80	0
Nodo 4	31	29	75	95	99	100	61	71	12	63	47	16	19	7	59	38	78	37	0	0	0	
Nodo 5	34	26	58	72	83	2	0	56	13	51	17	73	18	8	57	42	0	0				
Nodo 6	32	28	97	0	21	5		91	0	76	0	79	25	46	0	44						
Nodo 7	50	27	0		48	3		0		0		0	23	45		43						
Nodo 8	0	0			24	1							77	4		40						
Nodo 9					20	0							0	0		0						
Nodo 10					0																	

RC108

Número inicial de rutas = 19
Beta = 1,5
Gamma = 1

Número final de rutas = 20
Distancia total = 1940,00

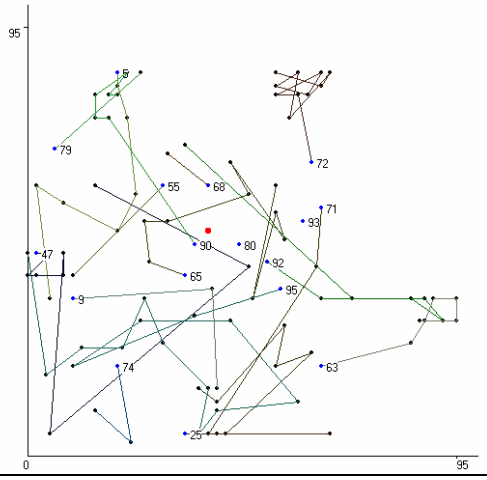


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20
Tiempo	189,1	225,31	218,62	217,63	200,34	173,01	169,59	179,73	217,8	193,98	132,51	219,6	198,16	193,34	176,68	185,71	194,57	167,06	195,41	182,29
Distancia	90,34	117,22	105,65	120,01	79,48	99,2	109,59	99,73	93,84	64,51	77,62	90,97	94,39	72,34	106,21	119,97	119,62	77,98	78,45	122,88
Carga	76	94	94	104	96	120	98	170	86	51	66	86	113	41	84	83	79	54	39	90
Espera	48,76	48,09	72,97	37,62	80,86	23,81	0	0	83,96	89,48	24,89	78,63	23,76	91	10,47	5,73	34,94	49,08	76,96	9,41
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	90	64	56	92	55	41	69	11	66	96	88	81	100	68	65	83	74	61	91	29
Nodo 3	99	20	51	50	6	39	98	15	22	93	60	54	2	42	52	24	58	80	84	28
Nodo 4	87	18	76	30	7	40	73	16	49	71	79	38	8	44	86	19	75	67	85	26
Nodo 5	59	21	89	32	4	43	78	47	25	72	0	37	46	0	82	48	97	94	63	27
Nodo 6	57	23	0	34	0	36	17	14	0	0		35	45		53	95	0	0	0	31
Nodo 7	0	77		33		0	13	12				0	3		70	62				0
Nodo 8		0		0			0	9					5		0	0				
Nodo 9								10					1							
Nodo 10								0					0							

RC201

Número inicial de rutas = 7
Beta = 1,81
Gamma = 1,03

Número final de rutas = 17
Distancia total = 2198,39



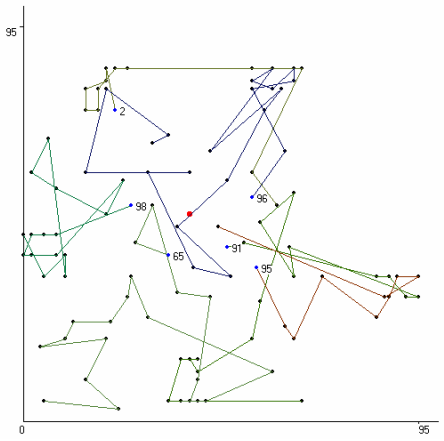
Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17
Tiempo	839,65	701,08	739,12	753,31	804,4	655,17	682,08	726,06	690,06	721,08	643,08	678,91	839,24	841,8	898,48	839,1	839,62
Distancia	150,09	160,54	153,65	289,47	224,4	151,12	139,96	193,3	99,14	102,04	121,69	74,05	40,64	111,12	129,76	42,19	15,23
Carga	130	126	208	334	166	164	96	112	116	50	46	46	24	35	55	3	13
Espera	619,56	460,54	475,47	313,84	490	404,05	472,12	442,76	520,91	599,03	491,39	584,86	778,6	700,68	738,72	786,9	814,38
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	5	72	95	47	65	63	71	90	79	9	55	68	74	25	93	80
Nodo 3	62	45	39	83	14	82	33	67	6	1	66	10	100	77	48	0	0
Nodo 4	31	2	42	59	16	98	30	21	7	0	20	0	0	58	89		
Nodo 5	28	88	36	52	15	69	29	18	8		0			0	0		
Nodo 6	34	53	38	64	11	81	27	85	3								
Nodo 7	50	78	44	76	12	61	26	51	46								
Nodo 8	70	73	40	19	75	94	32	84	4								
Nodo 9	0	13	41	23	91	96	0	49	0								
Nodo 10		0	35	22	60	56		24									
Nodo 11			37	57	0	54		0									
Nodo 12			43	99		0											
Nodo 13			0	86													
Nodo 14				87													
Nodo 15				97													
Nodo 16				17													
Nodo 17				0													

RC202

Número inicial de rutas = 6
Beta = 1,62
Gamma = 1

Número final de rutas = 6
Distancia total = 1435,10

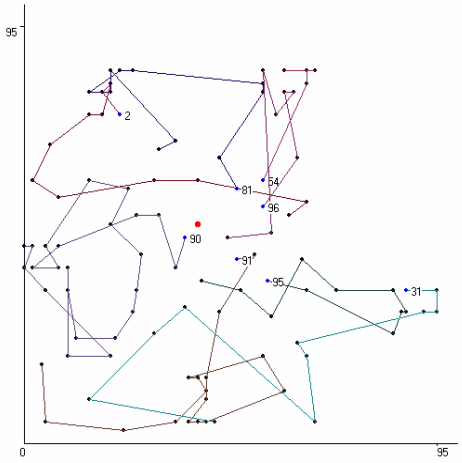
Nombre	R1	R2	R3	R4	R5	R6
Tiempo	888,48	839,10	853,71	753,31	897,75	839,62
Distancia	283,97	172,62	317,53	212,24	299,18	149,56
Carga	318	209	374	295	411	117
Espera	424,51	536,48	326,18	381,07	378,57	590,06
Nodo 1	0	0	0	0	0	0
Nodo 2	91	2	96	98	65	95
Nodo 3	92	5	72	12	82	85
Nodo 4	34	45	42	14	69	63
Nodo 5	31	8	37	47	83	50
Nodo 6	28	7	36	16	64	33
Nodo 7	26	6	39	15	19	30
Nodo 8	67	46	44	11	21	29
Nodo 9	62	3	40	88	23	27
Nodo 10	94	1	61	53	48	32
Nodo 11	71	43	38	78	18	80
Nodo 12	84	35	41	73	76	0
Nodo 13	51	54	81	79	57	
Nodo 14	49	93	90	9	99	
Nodo 15	22	0	56	10	52	
Nodo 16	20		66	13	86	
Nodo 17	24		55	17	87	
Nodo 18	25		68	0	59	
Nodo 19	89		60		97	
Nodo 20	0		4		74	
Nodo 21			70		58	
Nodo 22			100		77	
Nodo 23			0		75	
Nodo 24					0	



RC203

Número inicial de rutas = 8
Beta = 1,37
Gamma = 1,06

Número final de rutas = 8
Distancia total = 1392,95

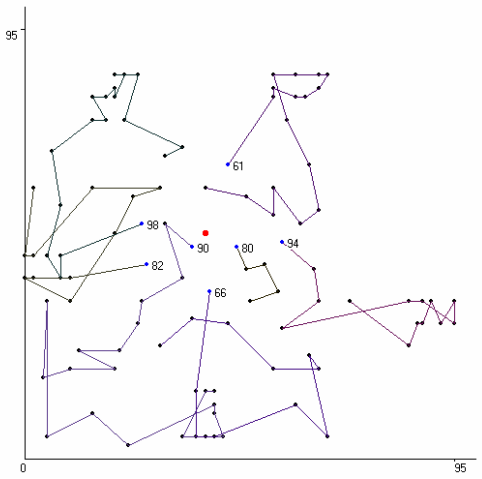


Nombre	R1	R2	R3	R4	R5	R6	R7	R8
Tiempo	915,48	839,10	853,71	770,98	711,20	839,62	599,26	876,61
Distancia	215,58	161,83	137,52	265,75	101,40	117,99	134,28	258,60
Carga	332	206	165	394	116	150	161	200
Espera	529,90	557,27	606,18	265,23	559,81	641,63	354,97	498,01
Nodo 1	0	0	0	0	0	0	0	0
Nodo 2	91	2	81	90	54	96	95	31
Nodo 3	92	46	61	65	37	72	62	29
Nodo 4	64	45	42	69	36	39	33	27
Nodo 5	20	6	44	98	40	38	32	26
Nodo 6	19	7	1	12	35	41	30	28
Nodo 7	23	79	3	15	0	43	34	85
Nodo 8	21	73	8	11		94	50	63
Nodo 9	48	78	4	14		80	67	89
Nodo 10	76	55	5	60		0	84	83
Nodo 11	51	68	70	88			56	57
Nodo 12	24	71	100	53			66	58
Nodo 13	22	93	0	82			0	18
Nodo 14	49	0		99				0
Nodo 15	25			52				
Nodo 16	77			86				
Nodo 17	75			87				
Nodo 18	97			9				
Nodo 19	0			10				
Nodo 20				59				
Nodo 21				74				
Nodo 22				13				
Nodo 23				16				
Nodo 24				47				
Nodo 25				17				
Nodo 26				0				

RC204

Número inicial de rutas = 7
Beta = 1,57
Gamma = 1,01

Número final de rutas = 7
Distancia total = 1092,69

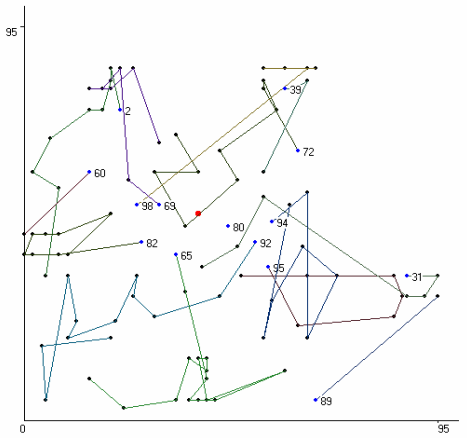


Nombre	R1	R2	R3	R4	R5	R6	R7
Tiempo	98,18	740,28	624,97	958,28	448,29	839,24	516,46
Distancia	48,18	177,78	163,37	236,06	168,85	156,77	141,68
Carga	65	191	233	363	213	352	307
Espera	0,00	442,50	321,59	522,21	149,44	502,47	194,78
Nodo 1	0	0	0	0	0	0	0
Nodo 2	80	82	94	90	66	98	61
Nodo 3	91	10	67	69	24	12	42
Nodo 4	92	15	62	65	23	11	44
Nodo 5	95	16	84	99	21	14	39
Nodo 6	56	9	34	52	48	78	38
Nodo 7	0	53	31	86	76	79	37
Nodo 8		88	26	87	89	7	35
Nodo 9		55	27	74	85	6	36
Nodo 10		60	28	59	63	8	40
Nodo 11		47	29	97	51	46	43
Nodo 12		17	30	13	64	45	41
Nodo 13		73	32	75	83	4	72
Nodo 14		0	33	58	57	3	71
Nodo 15			50	77	0	5	93
Nodo 16			0	49		1	54
Nodo 17				19		2	96
Nodo 18				18		70	81
Nodo 19				25		100	68
Nodo 20				22		0	0
Nodo 21				20			
Nodo 22				0			

RC205

Número inicial de rutas = 11
Beta = 1,57
Gamma = 1,05

Número final de rutas = 14
Distancia total = 1793,89

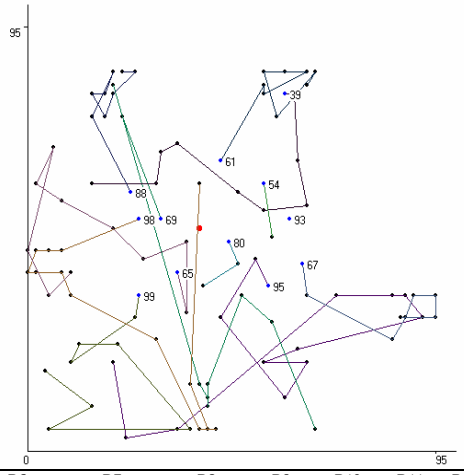


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
Tiempo	721,06	920,97	809,65	731,08	709,2	650,41	582,03	593,04	576,62	820,1	899,24	748,31	727,06	834,62
Distancia	193,08	191,77	149,52	141,13	131,43	122,16	83,85	143,81	112,15	167,57	105,2	88,45	148,54	15,23
Carga	211	288	133	172	119	77	66	125	164	125	139	37	55	13
Espera	407,97	579,19	560,12	499,95	527,77	468,26	458,17	369,23	374,47	562,53	714,04	639,86	558,52	809,38
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	65	72	2	98	95	39	31	82	94	69	60	89	80
Nodo 3	64	83	42	5	36	85	38	29	11	71	88	17	26	0
Nodo 4	57	19	44	45	40	33	37	27	15	62	3	0	0	
Nodo 5	52	23	41	6	43	32	54	28	16	63	46			
Nodo 6	99	21	61	7	35	34	0	30	47	50	8			
Nodo 7	86	18	81	79	0	56		96	14	67	4			
Nodo 8	59	76	90	73		0		91	12	84	1			
Nodo 9	87	48	55	78				66	53	51	100			
Nodo 10	9	22	68	13				0	10	93	0			
Nodo 11	75	20	70	0					0	0				
Nodo 12	97	49	0											
Nodo 13	74	24												
Nodo 14	0	25												
Nodo 15		77												
Nodo 16		58												
Nodo 17		0												

RC206

Número inicial de rutas = 9
Beta = 1,91
Gamma = 1,02

Número final de rutas = 12
Distancia total = 1669,88

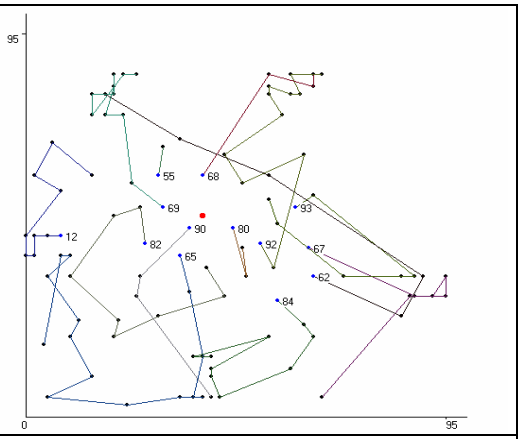


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
Tiempo	759,48	715,48	661,21	622,08	783,47	743,77	759,81	675,36	722,87	590,28	759,86	719,10
Distancia	259,35	230,96	126,82	135,02	151,82	188,56	174,76	117,33	160,29	47,31	35,47	42,19
Carga	220	229	169	102	157	179	243	169	172	39	42	3
Espera	360,13	384,52	444,39	407,05	531,65	465,21	445,05	468,02	452,59	522,97	694,38	666,90
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	95	69	61	67	39	99	98	88	65	54	80	93
Nodo 3	92	2	44	62	38	52	12	7	83	94	91	0
Nodo 4	64	45	42	33	72	59	14	5	90	0	66	
Nodo 5	76	22	36	30	71	87	47	46	82		0	
Nodo 6	63	49	40	29	96	86	16	8	53			
Nodo 7	51	19	43	27	81	23	15	6	78			
Nodo 8	85	20	41	26	70	75	11	4	73			
Nodo 9	28	56	35	32	100	58	9	1	79			
Nodo 10	31	84	37	0	55	97	57	3	17			
Nodo 11	34	89	0		60	0	21	0	13			
Nodo 12	50	0			0		18		10			
Nodo 13	25						48		0			
Nodo 14	77						24					
Nodo 15	74						68					
Nodo 16	0						0					

RC207

Número inicial de rutas = 10
Beta = 1,56
Gamma = 1

Número final de rutas = 13
Distancia total = 1599,36

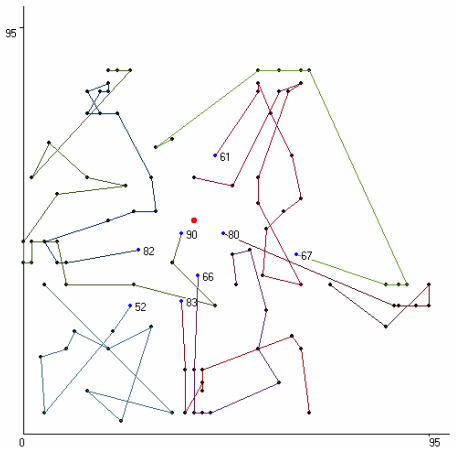


Nombre	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
Tiempo	555,27	668,49	563,52	604,17	657,06	639,08	515,09	677,38	649,93	691,04	635,45	673,04	467,31
Distancia	162,26	159,46	114,27	203,62	145,43	118,12	115,94	185,09	119,11	100,9	95,61	39,1	40,45
Carga	177	114	146	320	166	226	135	72	179	34	91	45	19
Espera	273,01	439,04	379,25	250,54	411,62	410,96	309,15	432,29	430,81	550,15	499,85	603,94	406,86
Nodo 1	0	0	0	0	0	0	0	0	0	0	0	0	0
Nodo 2	92	67	93	65	84	69	82	62	12	90	68	80	55
Nodo 3	95	30	71	83	85	88	98	33	14	99	43	56	100
Nodo 4	72	28	31	22	63	2	53	29	47	52	37	91	0
Nodo 5	81	27	34	23	76	6	9	54	15	48	36	0	
Nodo 6	61	26	50	21	18	45	86	70	16	0	0		
Nodo 7	41	32	94	25	19	5	74	46	17				
Nodo 8	42	89	96	77	49	4	57	0	78				
Nodo 9	44	0	0	75	51	8	64		73				
Nodo 10	39			58	24	7	66		79				
Nodo 11	38			59	20	3	0		60				
Nodo 12	40			87	0	1			0				
Nodo 13	35			13		0							
Nodo 14	0			10									
Nodo 15				11									
Nodo 16				97									
Nodo 17				0									

RC208

Número inicial de rutas = 8
Beta = 1,6
Gamma = 1

Número final de rutas = 8
Distancia total = 1402,78



Nombre	R1	R2	R3	R4	R5	R6	R7	R8
Tiempo	681,76	610,39	646,35	629,92	624,26	665,38	587,75	558,16
Distancia	235,85	197,22	129,96	128,96	229,77	153,41	154,02	173,59
Carga	202	229	133	210	312	286	147	205
Espera	325,91	243,17	436,38	390,96	214,49	361,97	343,73	284,57
Nodo 1	0	0	0	0	0	0	0	0
Nodo 2	52	61	80	66	90	82	83	67
Nodo 3	86	42	32	22	65	10	24	34
Nodo 4	75	44	30	21	64	11	23	31
Nodo 5	97	41	28	48	99	14	49	29
Nodo 6	59	72	26	18	9	53	19	35
Nodo 7	87	71	27	76	12	98	20	36
Nodo 8	74	93	33	51	47	69	85	40
Nodo 9	57	94	50	84	15	55	63	43
Nodo 10	77	95	0	92	16	2	89	100
Nodo 11	58	62		91	17	7	0	70
Nodo 12	25	96		56	78	46		0
Nodo 13	13	54		0	88	4		
Nodo 14	0	38			60	45		
Nodo 15		37			79	8		
Nodo 16		39			73	6		
Nodo 17		81			1	0		
Nodo 18		68			5			
Nodo 19		0			3			
Nodo 20					0			

AnexoV

Mejores Resultados de los problemas de Solomon (1987)

Fuente www.top.sintef.no/vrp/bknown.html
Fecha 5 Febrero, 2003

Problema	NV	Distancia	Autores	Problema	NV	Distancia	Autores
R101	19	1645.79	H	R201	4	1252.37	HG
R102	17	1486.12	RT	R202	3	1191.70	RGP
R103	13	1292.68	LLH	R203	3	939.54	M
R104	9	1007.24	M	R204	2	825.52	BVH
R105	14	1377.11	RT	R205	3	994.42	RGP
R106	12	1251.98	M	R206	3	906.14	SSSD
R107	10	1104.66	S97	R207	2	893.33	BVH
R108	9	960.88	BBB	R208	2	726.75	M
R109	11	1194.73	HG	R209	3	909.16	H
R110	10	1118.59	M	R210	3	939.34	M
R111	10	1096.72	RGP	R211	2	892.71	BVH
R112	9	982.14	GTA				
C101	10	828.94	RT	C201	3	591.56	RT
C102	10	828.94	RT	C202	3	591.56	RT
C103	10	828.06	RT	C203	3	591.17	RT
C104	10	824.78	RT	C204	3	590.60	RT
C105	10	828.94	RT	C205	3	588.88	RT
C106	10	828.94	RT	C206	3	588.49	RT
C107	10	828.94	RT	C207	3	588.29	RT
C108	10	828.94	RT	C208	3	588.32	RT
C109	10	828.94	RT				
RC101	14	1696.94	TBGGP	RC201	4	1406.91	M
RC102	12	1554.75	TBGGP	RC202	3	1367.09	CC
RC103	11	1261.67	S98	RC203	3	1049.62	CC
RC104	10	1135.48	CLM	RC204	3	798.41	M
RC105	13	1629.44	BBB	RC205	4	1297.19	M
RC106	11	1424.73	BBB	RC206	3	1146.32	H
RC107	11	1230.48	S97	RC207	3	1061.14	BVH
RC108	10	1139.82	TBGGP	RC208	3	828.14	IKMUY

Leyenda:

BVH - R. Bent y P. Van Hentenryck, "A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows," Technical Report CS-01-06, Department of Computer Science, Brown University, 2001.

BBB - J. Berger, M. Barkaoui y O. Bräysy, "A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows," Working paper, Defense Research Establishment Valcartier, Canada, 2001.

CC - Z. J. Czech y P. Czarnas, "A Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows," Proc. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, Canary Islands, Spain, (January 9--11, 2002), 376--383.

CLM - J.-F. Cordeau, G. Laporte, y A. Mercier, "A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows," Working Paper CRT-00-03, Centre for Research on Transportation, Montreal, Canada, 2000.

GTA - L. M. Gambardella, E. Taillard, y G. Agazzi, "MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows," in New Ideas in Optimization, D. Corne, M. Dorigo and F. Glover (eds), 63-76, McGraw-Hill, London, 1999.

HG - J. Homberger y H. Gehring, "Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows," ***INFOR***, VOL. 37, 297-318, (1999).

H - J. Homberger, "Verteilt-parallele Metaheuristiken zur Tourenplanung," Gaber, Wiesbaden (2000).

IKMUY - T. Ibaraki, M. Kubo, T. Masuda, T. Uno y M. Yagiura, "Effective Local Search Algorithms for the Vehicle Routing Problem with General Time Windows," Working Paper, Department of Applied Mathematics and Physics, Kyoto University, Japan, 2001.

LLH - H. Li, A. Lim, y J. Huang, "Local Search with Annealing-like Restarts to Solve the VRPTW," Working Paper, Department of Computer Science, National University of Singapore, 2001.

M - D. Mester, "An Evolutionary Strategies Algorithm for Large Scale Vehicle Routing Problem with Capacitate and Time Windows Restrictions," Working Paper, Institute of Evolution, University of Haifa, Israel (2002).

RT - Y. Rochat y E.D. Taillard, "Probabilistic Diversification and Intensification in Local Search for Vehicle Routing," ***Journal of Heuristics*** 1, 147-167, (1995).

RGP - L.M. Rousseau, M. Gendreau y G. Pesant, "Using Constraint-Based Operators to Solve the Vehicle Routing Problem with Time Windows," ***Journal of Heuristics,*** forthcoming.

SSSD - G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt y G. Dueck, "Record Breaking Optimization Results Using the Ruin and Recreate Principle," ***Journal of Computational Physics*** 159, 139-171, (2000).

S97 - P. Shaw, "A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems," Working Paper, University of Strathclyde, Glasgow, Scotland, 1997.

S98 - P. Shaw, "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems," In Principles and Practice of Constraint Programming - CP98, Lecture Notes in Computer Science, 417-431, M. Maher and J.-F. Puget (eds), Springer-Verlag, New York, 1998.