Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2021.

DIGITAL INTELLIGENCE – LINEAS DE ENSAMBLAJE

201800476 – Marvin Alexis Estrada Florian

Resumen

Los robots ayudan a la eficacia del desarrollo del hombre en diferentes desgloses de su vida cotidiana, hoy en día tenemos robots que nos ayudan a hacer tareas básicas en el área profesional e investigación de alto presupuesto, por lo que las personas involucradas en cada proyecto buscan obtener la mayor optimización de recursos que se les proporciona de parte de inversores individuales o corporaciones para que ese proceso se lleve a cabo y generar resultados favorables. Algunas áreas de manufactura en el mundo actual están totalmente automatizadas por robots las cuales son fábricas de producción masiva que necesitan la optimización de sus recursos y tiempo, el cual es uno de los factores más importantes a la hora de producir productos a gran escala. Los procesos de fabricación se ven claramente afectados al no poseer un procedimiento de ensamblaje optimizado, para esto se conocen las líneas de ensamblaje, creadas y optimizadas en el siglo pasado haciendo la propuesta que un producto puede ser construido a la vez por áreas y así optimizar el tiempo de construcción como recursos de la empresa.

Palabras clave

Robot, ensamblaje, recursos, consumo, optimización.

Abstract

Robots help the effectiveness of man's development in different breakdowns of his daily life, today we have robots that help us to do basic tasks in the professional area and high-budget research, so the people involved in each project seek obtain the best optimization of resources that are provided to them by individual investors or corporations so that this process is carried out and generate favorable results. Some manufacturing areas in the world today are fully automated by robots which are mass production factories that need optimization of their resources and time, which is one of the most important factors when producing large-scale products. The manufacturing processes are clearly affected by not having an optimized assembly procedure, for this the assembly lines, created and optimized in the last century are known, making the proposal that a product can be built at the same time by areas and thus optimize construction time as company resources.

Keywords

Robot, assembly, resources, consumption, optimization.

Introducción

La empresa Digital Intelligence, S. A. ha desarrollado una máquina capaz de ensamblar las partes de cualquier producto.
La máquina creada por Digital Intelligence, S.A. puede construir cualquier producto ensamblando automáticamente los componentes (partes) que lo conforman. Para esto, la máquina desarrollada consta de n líneas de ensamblaje y un brazo robótico para cada una de éstas, además, cada línea de ensamblaje posee un mecanismo que le permite acceder a m componentes distintos. El brazo robótico demora 1 segundo en colocarse sobre el recipiente que contiene el 1er componente, 2 segundos para colocarse en el recipiente que contiene el 2do componente y así sucesivamente hasta requerir m segundos para colocarse en el m-ésimo componente que se le solicite. Adicionalmente, para ensamblar el componente en el producto que se construye, el brazo robótico utilizará Xm segundos para el m-ésimo componente.

Algoritmo de construcción con robots en líneas de ensamblaje

El desarrollo de este algoritmo se compone de varias partes pero sin embargo la característica principal de que funcione son las listas enlazadas, las cuales es una de las estructuras de datos fundamentales, y puede ser usada para implementar otras estructuras de datos. Consiste en una secuencia de nodos, en los que se guardan campos de datos arbitrarios y una o dos referencias, enlaces o punteros al nodo anterior o posterior. El principal beneficio de las listas enlazadas respecto a los vectores convencionales es que el orden de los elementos enlazados puede ser diferente al orden de almacenamiento en la memoria o el disco, permitiendo que el orden de recorrido de la lista sea diferente al de almacenamiento. (Paul E.Black, 2004).

Ya que se conoce lo que es una lista enlazada procedemos a guardar los datos del archivo de entrada en ella ya que nos servirá para poder reconocer cuantas líneas de ensamblaje hay, como también que productos podremos ensamblar como también sus pasos de ensamblaje respectivamente para que este proceso sea optimizado y ahorrar tiempo de construcción y ensamblaje. En la figura I podemos observar como los datos son proporcionados y el algoritmo hace su tarea que es encontrar el camino óptimo (marcado en color gris) y llegar a la posición final. Para hacer funcionar este algoritmo necesitamos dos archivos de entrada, lo que es el archivo de máquina, el cual como se menciona anteriormente, contiene la cantidad de líneas de ensamblaje que contendrá nuestro entorno como su número de identificación, cantidad de componentes que posee y el tiempo que demora en ensamblar, así para cada una de las líneas de ensamblaje en el listado, también este archivo posee un listado de productos en el cual cada uno tiene como atributos su nombre y los pasos que se deben de seguir para su ensamblaje, estos pasos tienen un formato específico, ya que empieza con la letra L, la cual nos referencia a un carácter siguiente que es el número de línea de ensamblaje que debe de ensamblar ese componente, seguido de la letra C, el cual nos menciona con un carácter siguiente, el número de componente que se tiene que ensamblar, dando como resultado un paso de ensamblaje ejemplo “L1C1”. Estos pasos de ensamblaje no vienen directamente de esta forma en el archivo de entrada sino que vienen acompañados de dos “p” las cuales

mediante expresiones regulares son retiradas para poder acceder a la información necesitada para realizar el proceso de ensamblaje de un producto determinado, como se muestra en la Figura I.

```
<Maquina>
    <CantidadLineasProduccion>
        <!--[0<NumeroEntero<1000]-->
    </CantidadLineasProduccion>
    <ListadoLineasProduccion>
        <LineaProduccion>
            <Numero>
                <!--[NumeroEntero>0]-->
            </Numero>
            <CantidadComponentes>
                <!--[0<NumeroEntero<1000]--> *Componentes de la línea
            </CantidadComponentes>
            <TiempoEnsamblaje>
                <!--[NumeroEntero>0]--> *Representa el tiempo en segundos
                                         que toma ensamblar en la línea
            </TiempoEnsamblaje>
        </LineaProduccion>
        …
    </ListadoLineasProduccion>
    <ListadoProductos>
        <Producto>
            <nombre>
                <!--[Texto]--> *Nombre del Producto
            </nombre>
            <elaboracion> *Corresponde a instrucciones para ensamblar el producto
                <!--[Texto con formato L1pC1p L2pC2p L3pC3p … LnpCnp]-->
            </elaboracion>
        </Producto>
        …
    </ListadoProductos>
</Maquina>
```

Figura 1. Archivo de entrada de máquina.

Fuente: Elaboración propia, Septiembre 2021.

Como se menciona anteriormente se necesitan de dos archivos de entrada para hacer funcionar el algoritmo de ensamblaje de un producto, con el primero se tiene la carga de información de lo que tenemos a disposición para ensamblar, líneas de ensamblaje, productos, etc. Mientras que en el segundo archivo tenemos los datos de simulación, los cuales se conforman de un listado de productos los cuales son los que se van a ensamblar específicamente, seguido del nombre de la simulación que ha sido cargada, esto para poder identificar que simulación se realizó cuando se generen los reportes y tener un mejor almacenamiento de los procesos introducidos en el algoritmo, como se muestra en la Figura II.

```
<Simulacion>
    <Nombre>
        <!--[Texto]-->
    </Nombre>
    <ListadoProductos>
        <Producto>
            <!--[Texto]-->
        </Producto>
        <Producto>
            <!--[Texto]2-->
        </Producto>
        …
    </ListadoProductos>
</Simulacion>
```

Figura II. Archivo de entrada de simulación.

Fuente: Elaboración propia, Septiembre 2021.

Como se mencionó anteriormente se tienen dos cargas de datos para obtener resultados del algoritmo, a partir de esto se hace una inicialización del proceso de ensamblaje descrito en cada uno de los productos cargados a través del archivo de simulación, se empieza moviendo todos los brazos robóticos hacia el componente uno y luego irlos moviendo como se solicita. Los datos de entrada se cargan mediante en un archivo XML el cual es un metalenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consorcio, utilizado para almacenar datos en forma legible. Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos, de la misma manera que HTML es a su vez un lenguaje definido por SGML, para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información. XML no ha nacido únicamente para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de

cálculo y casi cualquier cosa imaginable. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande, con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil (Abraham Silberschatz, 2012). La aplicación mediante estos archivos funciona de la siguiente manera, en una simulación de ejemplo que se tengan dos líneas de ensamblaje, cada una se demora un segundo en ensamblar un componente, para lo cual se inicia el proceso moviendo los brazos robóticos hacia el primero de estos, se inicia un bucle en el cual se valida si la línea actual por ejemplo la uno, se encuentra entre los pasos descritos del producto a ensamblar, si encuentra alguna coincidencia se valida si ese componente no ha sido ensamblado antes, para lo cual se tiene un booleano que inicial en False en todos los pasos de ensamblaje y a medida que cada uno se va ensamblando este cambia a True respectivamente, luego de esta validación se verifica si en ese segundo específico hay otro componente ensamblándose, debido a que se solicita que solo un brazo robótico debe de poder estar ensamblando un componente a la vez, si después de todas estas validaciones da negativo a todas, se procede a seguir las siguientes validaciones, si el componente que se solicita ensamblar es mayor o menor al componente en el que se encuentra la línea de ensamblaje actual, entonces le suma o resta en uno a la posición del componente de esa línea en específico debido a cual sea el caso, si es mayor le suma uno, si es menor le resta uno, así segundo a segundo se va acercando al objetivo solicitado. Si el componente en el que se encuentra actualmente la línea de ensamblaje es igual al componente solicitado y pasó con éxito todas las validaciones mencionadas anteriormente entonces, si y solo si empieza a ensamblar, este proceso tarda el tiempo cargado en el archivo de máquina de entrada, para este ejemplo pues únicamente tarda un segundo, y como se menciona anteriormente, en este segundo específico no puede ensamblar otra línea de ensamblaje en el cual su componente actual sea el solicitado, media vez termine el proceso de ensamblaje que está ocurriendo en ese momento se puede proceder a ensamblar la siguiente. Así se va desarrollando el algoritmo realizando una serie de validaciones para poder llegar a su objetivo que es, ensamblar todos los productos solicitados.

Luego de este proceso el cual es el principal de todo el programa se puede proceder a realizar lo que son los archivos de salida o reportes de este proceso de ensamblaje de productos, estos se dividen en tres áreas, reportes HTML, XML y Graphviz. A continuación en la Figura III, se nuestra un ejemplo de un archivo de salida o reporte en HTML de un proceso de ensamblaje de un producto llamado Smartwatch.

Tiempo	Línea de ensamblaje 1	Línea de ensamblaje 2
1er. Segundo	Mover brazo – componente 1	Mover brazo – Componente 1
2do. Segundo	Mover brazo – componente 2	No hacer nada
3er. Segundo	Ensamblar componente 2	No hacer nada
4to. Segundo	Mover brazo – Componente 3	Ensamblar – Componente 1
5to. Segundo	Mover brazo – Componente 4	Mover brazo – Componente 2
6to. Segundo	No hacer nada	Ensamblar – Componente 2
7mo. Segundo	Ensamblar componente 4	No hacer nada
El producto SmartWatch se puede elaborar óptimamente en 7 segundos.		

Figura No. 2 - Tabla de Resultados

Figura III. Salida como archivo HTML de ejemplo.

Fuente: Elaboración propia, Septiembre 2021.

Como también otra forma de visualización de resultados se hará un reporte por medio de la herramienta Graphviz, que es un programa de visualización gráfica de fuente abierta. Esta representa la visualización de gráficos como una forma de representar información estructural como diagramas de gráficos y redes abstractos. Tiene importantes aplicaciones en redes, bioinformática, ingeniería de software, diseño de bases de datos y web, aprendizaje automático y en interfaces visuales para otros dominios técnicos. Los programas de diseño Graphviz toman descripciones de gráficos en un lenguaje de texto simple y crean diagramas en formatos útiles, como imágenes y SVG para páginas web; PDF o Postscript para su inclusión en otros documentos; o mostrarlo en un navegador de gráficos interactivo. Graphviz tiene muchas características útiles para diagramas concretos, como opciones de colores, fuentes, diseños de nodos tabulares, estilos de línea, hipervínculos y formas personalizadas.

En la práctica, los gráficos generalmente se generan a partir de fuentes de datos externas, pero también se pueden crear y editar manualmente, ya sea como archivos de texto sin formato o dentro de un editor gráfico. Graphviz no estaba destinado a ser un reemplazo de Visio, por lo que probablemente sea frustrante intentar usarlo de esa manera (AT&T labs research, 1991). Por lo tanto, como se mencionó anteriormente, la salida consiste en una serie de 2 pasos, se le solicita al usuario seleccionar el nombre del producto del cual desea generar el reporte, luego del segundo específico del cual se desea ver el proceso de ensamblaje que está sucediendo en ese momento, para la visualización en graphviz se muestran los pasos de ensamblaje, cada uno de un color determinado, si el paso es verde quiere decir que ya fue ensamblado, si es rojo quiere decir que aún falta de ensamblarlo, también se muestra el nombre del producto seleccionado y en qué segundo se solicitó la generación del reporte. Se muestra a continuación un ejemplo de la cola de un producto en un segundo específico figura IV.

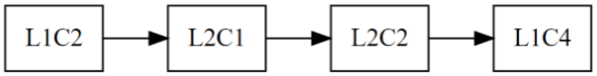

Figura IV. Salida como archivo de Graphviz en PDF de ejemplo.

Fuente: Elaboración propia, Septiembre 2021.

Todo esto es debido a la utilización de listas enlazadas, que permiten la optimización en la utilización de la memoria a largo plazo y sobre todo en métodos y procesos grandes para poder realizar tareas no solo efectivas sino de la forma correcta que es optimizando la memoria de nuestro dispositivo. Esto se hace a través de nodos almacenados en estas listas que pueden contener cualquier tipo de dato en cada uno de estos, haciendo estas listas de las mejores opciones para guardar datos. Para lo que los métodos de búsqueda de posiciones se basan en el almacenamiento de los datos obtenidos de los archivos XML. Para esto también se utilizaron listas circulares, las cuales permiten la iteración infinita que sirvió para ir comparando las listas cargadas junto con las solicitadas en los pasos de ensamblaje de los productos cargados mediante los archivos de entrada. La lista circular es una especie de lista enlazada simple o doblemente enlazada, pero que posee una característica adicional para el desplazamiento dentro de la lista: esta no tiene fin. Para que la lista no tenga fin, el puntero siguiente del último elemento apuntará hacia el primer elemento de la lista en lugar de apuntar al valor NULL, como hemos visto en el caso de listas enlazadas simples o doblemente enlazadas.

En las listas circulares, nunca se llega a una posición en la que ya no sea posible desplazarse. Cuando se llegue al último elemento, el desplazamiento volverá a comenzar desde el primer elemento. A continuación, veamos el algoritmo de inserción y el registro de los elementos: declaración del elemento que se insertará, asignación de la memoria destinada al nuevo elemento, llenar el contenido del campo de los datos, actualizar los punteros hacia el primer y último elemento si fuese necesario.
Caso particular: en una lista con un solo elemento, el primer elemento es al mismo tiempo el último. Luego, se tiene que actualizar el tamaño de la lista.
(Jean-Francois Pillou, 2020).

Conclusiones

Se concluyó que la optimización de algún proceso en la vida cotidiana es posible mediante la implementación de algún algoritmo determinado, en este caso la construcción de productos mediante pasos en líneas de ensamblaje.

Se determinó que el uso de listas enlazadas como también listas circulares para el almacenamiento de datos es bastante útil y luego para solicitarlos y utilizarlos según sea necesario para complementar la lógica de sus uso con el del algoritmo.

Concluyendo la utilización de HTML y Graphviz como medio de representación gráfica de datos resulta bastante útil para enseñarle al usuario los resultados del algoritmo implementado y pueda llegar a una conclusión como objetivos más fácilmente.

Anexos

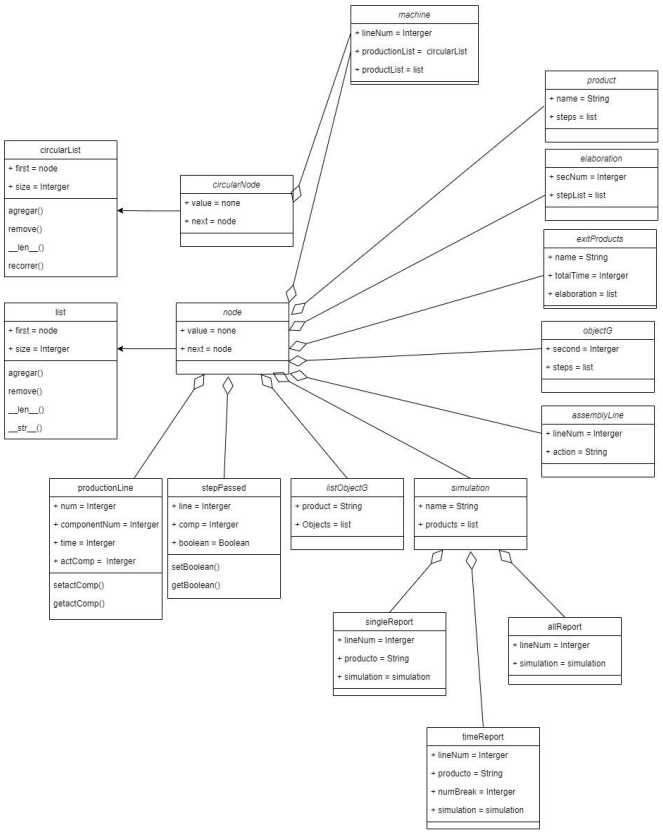

Figura VII. Diagrama de clases del programa.

Fuente: Elaboración propia, Septiembre 2021.

Referencias bibliográficas

Paul E.Black, (2004). *Definition of a linked list.* National Institute of Standards and Technology.

Abraham Silberschatz, (2012). *Fundamentos de bases de datos,* McGraw-Hill.

Jean-Francois Pillou, (2020). *Listas circulares,* CCM.

AT&T labs research, (1991). *What is graphviz,* graphviz org.