Tesis Joshi…

**Clustering Poligonal Restringido**

Es posible considerar el problema de distritado como un problema de conglomeración (*clustering*) en el que se requiere conglomerar todo el conjunto de polígonos en grupos de manera tal que se maximice una dada función predefinida (Altman 2001).

Generalmente los algoritmos de conglomeracion propuestos en la literatura se focalizan en datos de puntos en lugar de polígonos (ver por ejemplo Han, Kamber & Tung 2001) y cuando se aplican estos algoritmos diseñados para conglomerar puntos con un conjunto de datos poligonales, los resultados no son buenos (Joshi, Samal & Soh, 2009), básicamente por los diferentes tipos de relaciones topológicas que pueden darse entre polígonos en relación con los puntos.

El algoritmo propuesto por Joshi (XXXX tesis) es el primero según nuestra búsqueda que se aplica a polígonos en lugar de a puntos.

Las restricciones aplicadas durante el proceso de conglomeración pueden ser de dos tipos:

1. **Restricciones al nivel de la instancia**
2. **Restricciones al nivel del conglomerado**.

Las restricciones al nivel de la instancia se aplican a los objetos individuales que se están conglomerando, por ejemplo las restricciones del tipo “deben ir juntos” (*must link*) y “no pueden ir juntos” (*cannot link*) (Davidson y Ravi 2005) (XXXXXXX BUSCAR XXXXXXXX). Por otro lado, las restricciones al nivel del conglomerado se aplican justamente a propiedades de todo el conglomerado (por ejemplo las restricciones de promedio o de suma (cantidad total de población del conglomerado ≤ x)). Altman (2001) ha probado que satisfacer este tipo de restricciones es un problema NP-difícil.

Joshi (XXX TESIS), presenta un conjunto de algoritmos de conglomeración para polígonos en presencia de restricciones. El algoritmo central se denomina “***Constrained Polygonsal Spatial Clustering” (CPSC)*** (Conglomeración Poligonal Espacial Restringida), este algoritmo CPSC está diseñado para resolver problemas en los que las restricciones son fuertes e inviolables, y luego sobre este algoritmo base presentan modificaciones que permiten suavizar las restricciones, o subdividir las unidades base durante el proceso de conglomeración permitiendo otras variantes del algoritmo.

CPSC emplea un proceso de búsqueda del tipo A\*, y se divide en tres pasos principales: 1) seleccionar las semillas, 2) decidir cuál es el mejor conglomerado para crecer y 3) elegir el mejor polígono para ser añadido al mejor conglomerado. Algunas de las estrategias innovadoras de CPSC incluyen:

* *Empleo de una función heurística para aplicar las restricciones durante la conglomeración*: La función heurística (F) tiene dos componentes: (1) la función de distancia (*H*) que mide la distancia entre el estado actual del conglomerado con el estado objetivo, y (2) la función de costo (*C*) que mide la reducción en la flexibilidad del crecimiento de todos los restantes conglomerados.
* *Integración de las restricciones en la selección de las semillas*: Se emplean desde el inicio restricciones al nivel de la instancia y al nivel del conglomerado para la selección de las semillas. Al aplicar las restricciones y seleccionar las semillas empleando las funciones heurísticas, se hace al algoritmo más robusto en relación con la elección de semillas.
* *Selección del mejor conglomerado para crecer*: Al comienzo de cada iteración el CPSC selecciona el mejor conglomerado para crecer basándose en la heurística (*F*) que aproxima el nivel de “*necesidad de crecer*” de cada conglomerado, y por lo tanto el conglomerado con la mayor necesidad es el que se elige para crecer.
* *Selección del mejor polígono para ser añadido al mejor conglomerado*: Una vez que el mejor conglomerado ha sido seleccionado para crecer, se elige el mejor polígono en términos del nivel de “reducción de flexibilidad” empleando la función heurística *F*’; es decir, el polígono con el menor impacto en el crecimiento de los restantes conglomerados se elige para ser añadido al mejor conglomerado.
* *Se permite a las unidades base (polígonos) moverse de un conglomerado a otro*: Dado que el crecimiento de un conglomerado sigue una estrategia voraz (*greedy*), cada conglomerado selecciona el polígono que minimiza su necesidad en esa etapa. Un conglomerado puede decidir que un polígono que ya ha sido asignado a otro conglomerado es el mejor polígono para sus necesidades. Mover un polígono de su conglomerado original a un nuevo conglomerado, está permitido por el algoritmo CPSC en el caso especial cuando un nuevo conglomerado no tiene ningún polígono no asignado (libre) en su vecindad.

En nuestro caso aplicaremos la variante CPSC\* del algoritmo de conglomeración. Es importante resaltar que la familia de algoritmos CPSC presenta ciertas ventajas en relación con los algoritmos de redistritado usualmente empleados, ya que no requiere un plan inicial (una solución inicial) donde *cada* polígono ha sido asignado a un distrito (conglomerado / *cluster*), por el contrario, la familia de algoritmos CPSC selecciona las semillas entre las unidades base y luego crece los conglomerados a partir de estas semillas. Por otro lado, la familia CPSC también define una metodología clara para la selección de las semillas ya que se basa en un conjunto pre-definido de restricciones en lugar de la mera selección aleatoria de semillas que emplean los algoritmos de partición de grafos. Finalmente, dado que la familia CPSC sigue una búsqueda del tipo A\* para el crecimiento de los conglomerados, no hay riesgo de quedar atrapados en un óptimo local.

**Algoritmos CPSC (Joshi, TESIS)**

El objetivo central de los algoritmos CPSC es crecer conglomerados que satisfagan las restricciones del problema en el que se aplican, el algoritmo está diseñado para inherentemente producir conglomerados espacialmente contiguos y compactos. A este fin, se emplean las nociones de *restricciones a nivel de la instancia* y *restricciones a nivel del conglomerado*. El algoritmo de búsqueda que subyace en esta familia es como ya se ha dicho del tipo A\* (Russell & Norvig 2003).

**Preliminares:**

**Algoritmo de Busqueda A\*:** es un algoritmo del tipo “*best-first search*” que encuentra el camino menos costoso a partir de un nodo inicial hacia un nodo final. Emplea una función heurística (*F(n) = G(n) + H(n))* que es una combinación de una función de costo del camino (*G(n))* y una función admisible de distancia (*H(n)*) (en el sentido de una función de distancia que no sobre estime la distancia al objetivo. La función de costo del camino *G(n)* mide el costo de arribar al nodo actual desde el nodo inicial, y la función de distancia *H(n)* mide la distancia estimada desde el nodo actual al nodo objetivo.

Empezando con el nodo inicial, A\* mantiene una cola prioritaria de nos nodos a ser recorridos, conocida como el conjunto abierto o simplemente OPEN. Cuanto menor sea *F(n)* para un dado nodo *n*, mayor es su prioridad. En cada paso del algoritmo, el nodo con el menor valor de *F(n)* se remueve de la cola OPEN y se añade a otra cola conocida como conjunto cerrado o CLOSED. Los valores de *F* y *H* de sus vecinos se actualizan y aquellos vecinos que no han sido ya añadidos a OPEN o a CLOSED se añaden a la cola OPEN. El algoritmo continúa hasta que el nodo objetivo se alcanza (o hasta que la cola OPEN está vacía). El valor *F* del objetivo es entonces la longitud del camino más corto (Rusell & Norvig, 2003). El esqueleto del algoritmo es simplemente:

1. Comenzar en el nodo inicial n0
2. Colocar n0 en una cola denominada OPEN
3. Crear una cola denominada CLOSED inicialmente vacía.
4. Si OPEN es vacío, salir indicando fracaso.
5. Remover de OPEN el nodo *n* que tiene el menor *F(n)* y ponerlo en CLOSED.
6. Si *n* es el nodo objetivo, salir indicando éxito.
7. Expandir el nodo *n*, generando el conjunto *M* de sus vecinos.
8. Agregar a OPEN a los miembros de *M* que no se encuentren ya en OPEN o en CLOSED.
9. Reordenar la lista OPEN por valores crecientes de *F*.
10. Ir al paso 5

**Contigüidad Espacial:** Un conglomerado de polígonos es espacialmente contiguo cuando cada polígono dentro del mismo comparte por lo menos una parte de su frontera con por lo menos otro polígono dentro del conglomerado

**Compacidad del conglomerado:** El circulo es la figura más compacta para cualquier conglomerado dado que cubre la mayor superficie con el menor perímetro (Clayton 2000 BUSCAR). XXXXXX REDIRIGIR A LAS METRICAS DE COMPACIDAD…

**Tipos de Restricciones:** En muchos casos existe cierto conocimiento del dominio del problema y por lo tanto en lugar de emplear este conocimiento con fines de validación, puede ser empleado como guía o ajuste de un proceso (conglomeración) que es por definición no supervisado. La aproximación resultante se conoce como conglomeración semi-supervisada o conglomeración restringida (Basu, Banerjee y Mooney, 2002 y XXXXX AGREGAR LOS TRES DE SUGATU). La conglomeración restringida hace uso del conocimiento del dominio transformándolo en un conjunto de restricciones que son aplicadas durante el proceso de agrupamiento de objetos de datos que se están conglomerando. Como ya hemos mencionado, las restricciones aplicadas durante el proceso de conglomeración pueden ser de dos tipos, (1) Restricciones al nivel de la instancia y (2) Restricciones al nivel del conglomerado.

Las **restricciones al nivel de la instancia** se aplican a los objetos individuales que se están conglomerando. Existen básicamente dos tipos de restricciones al nivel de la instancia, las restricciones *deben ir juntos* (*must-link*) y las restricciones *no pueden ir juntos* (*cannot-link*) (Davidson y Ravi, 2005 XXXX BUSCAR). DAR EJEMPLOS DE ESTAS RESTRICCIONES.

Las **restricciones al nivel del conglomerado** se aplican al cluster como un todo, por ejemplo restricciones de suma o de promedio (Davidson y Ravi, 2004, XXXX BUSCAR). DAR EJEMPLOS DE ESTAS RESTRICCIONES.

**Funciones heurísticas basadas en restricciones**: Para poder incorporar los diferentes tipos de restricciones dentro del proceso de conglomeración, se toma prestada la idea de una función heurística *(F)*, de los algoritmos heurísticos de búsqueda. *F* es una combinación de:

1. Una función que aproxima la distancia del estado actual del cluster al estado ideal *(H)* midiendo por lo tanto el nivel de necesidad del cluster de crecer aún más, y
2. Una función de costo que mide la reducción en flexibilidad en el crecimiento de los restantes clusters *(G)*.

Empleando lo anterior, *F* se define como una suma de los dos, es decir

*F = H + G* (1)

La función de distancia *H* toma en cuenta las restricciones a nivel del conglomerado para encontrar la distancia entre el estado actual del cluster y su estado objetivo (target), mientras que la función de costo *(G)* toma en cuenta el efecto del crecimiento de cada cluster sobre los otros clusters. De esta forma es posible elegir que cluster crecer y que unidad base (polígono) anexar a ese cluster elegido. La reducción en flexibilidad en el crecimiento de los clusters se considera una función de costo dado que una elección basada solamente en *H* puede tener un efecto adverso sobre el crecimiento de los restantes clusters. Con la adición de *G*  a *H* penalizamos un nodo si restringe el crecimiento de otros nodos, y por lo tanto se evita que el algoritmo CPSC sea meramente voraz (*greeedy*).

Para seleccionar la función de distancia *H* el primer paso es identificar cuáles son las restricciones que se cuantifican más fácilmente y cuáles son las más importantes para satisfacer en nuestro problema. Por ejemplo en el caso de los distritos electorales, la restricción más importante es la igualdad poblacional dentro de un dado margen de error, al igual que la contigüidad espacial. Mientras que otra restricción importante es la compacidad. Por lo tanto consideramos nuestros conglomerados objetivos como aquellos que son espacialmente contiguos y compactos con igualdad de población.

Una vez que se han definido las propiedades deseables de los conglomerados objetivo, para poder definir operativamente a *H* necesitamos identificar cuales restricciones son aplicables al nivel del conglomerado. Por ejemplo, en nuestro caso, las restricciones de igualdad de población y compacidad espacial son restricciones a nivel del conglomerado, mientras que la contigüidad espacial puede ser fácilmente traducida en restricciones a nivel de la instancia (del tipo ya mencionado “deben ir juntos”, “no pueden ir juntos”, etc.). Por lo tanto, si empleamos las restricciones a nivel del conglomerado que nos permiten arribar al estado objetivo, la función de distancia *H* puede ser definida como:

(2)

donde es la distancia entre el estado actual del cluster y el estado objetivo para ese mismo cluster basándose en la x-esima restricción a nivel del conglomerado.

Que para el caso electoral, considerando solamente población y compacidad se traduce como:

(3)

donde depende del conocimiento del problema en relación a la población total de cada conglomerado. Por otro lado, la restricción de compacidad fuerza a que el conglomerado crezca para formar el distrito más compacto posible. Dado que el circulo es la forma más compacta, si se empleara el índice de Schwartzberg, por ejemplo (Schwartzberg, 1996) el valor para el circulo (=) es de 4π, mientras que y son atributos del estado actual del conglomerado.

Con el uso de la función de costo *G* el objetivo es seleccionar el conglomerado a crecer de manera tal que preserve el máximo grado de flexibilidad para el crecimiento de los otros conglomerados. Para cumplir con este objetivo, es necesario observar el efecto del crecimiento de un conglomerado sobre la habilidad de crecimiento de los restantes. Esta función está directamente asociada con restricciones independientes del dominio, como la de asignar cada unidad base (polígono) a un conglomerado y que estos conglomerados sean espacialmente contiguos y compactos. Un ejemplo de función de costo *G* es:

(4)

donde *k* es la cantidad de conglomerados, *n* es la cantidad de unidades base (polígonos) que rodean un conglomerado (es decir que son sus vecinos) y que no han sido asignados aún a ningún conglomerado, *O(Ci)* es la frontera exterior de un conglomerado i (asumiendo que todos los polígonos dentro del conglomerado son contiguos) que es compartida con polígonos que aún no han sido asignados a ningún otro conglomerado, y *O’(Ci,j)* es la nueva frontera resultante para el conglomerado *i* luego de añadirle el conglomerado *j*. Intuitivamente esta función de costo dice que si añadir un nuevo polígono hace más compacto al conglomerado, entonces la función de costo es negativa; de otro modo la función de costo es positiva en la medida en que el conglomerado crece agresivamente reduciendo la flexibilidad de crecimiento de otros conglomerados que también se beneficiarían mucho con la incorporación de nuevas unidades base (polígonos). Esta función de costo promoverá un crecimiento paralelo de los conglomerados.

Otro ejemplo de función de costo *G* es:

(5)

Donde *k* es la cantidad de conglomerados, *n* es la cantidad de unidades base que rodean un conglomerado y que no han sido aún asignados a ningún conglomerado, *N(Ci)* es la cantidad de vecinos libres que tiene el conglomerado *i*, y *N’(Ci,j)* es la cantidad de vecinos libres del conglomerado *i* luego de haberle añadido el nuevo polígono *j*. Intuitivamente esta función de costo lleva a una carrera para la conglomeración completa y alienta a un conglomerado a dominar el proceso de conglomeración al premiarlo por añadir polígonos que le den mayor cantidad de vecinos libres. Esta función de costo promoverá un crecimiento secuencial de los conglomerados.

Si *H* sobrestima la distancia del estado actual del conglomerado en relación con el estado buscado (objetivo / target), el proceso de conglomeración no saltará de un conglomerado a otro, sino que crecerá de a un conglomerado, y por lo tanto el proceso será secuencial. Por otro lado, si *H* subestima la distancia, entonces el proceso de conglomeración será sensiblemente más lento. En forma análoga, una función de costo más astringente provocará un proceso de conglomeración más lento con cada conglomerado seleccionando un polígono para crecer en forma muy conservadora.

Joshi en su tesis al ejemplificar el caso del distritado electoral en el que la contigüidad espacial y la compacidad son importantes, elije la función de costo basándose en la longitud de la frontera común de los conglomerados, definida por la ecuación (4), dado que esta función de costo penaliza en forma máxima a los conglomerados si no son espacialmente compactos.

**El Algoritmo CPSC**

El algoritmo CPSC comienza seleccionando las semillas en el conjunto de unidades base (polígonos). Dado que cada semilla va a crecer para convertirse en un conglomerado, cada semilla representa justamente un futuro conglomerado y por lo tanto el mecanismo de selección sigue una lógica contra-intuitiva en la que cada polígono semilla debe violar todas las restricciones “*deben ir juntos*” con relación a los demás polígonos semilla. De lo contrario las semillas resultantes pueden conglomerarse dentro del mismo conglomerado, haciendo que la selección de semillas sea inválida. Para esto, las semillas se seleccionan empleando una búsqueda sistemática basándose en el conocimiento que se tiene del dominio. De esta forma, se miden las heurísticas dependientes del dominio para cada polígono, por ejemplo, la distancia entre dos polígonos, la población de cada uno, la superficie de cada uno, etc. Luego en base a las propiedades buscadas para los conglomerados objetivo, se seleccionan las restricciones más importantes y la propiedad correspondiente se implementa y calcula para cada polígono. Por ejemplo, en nuestro caso, la restricción que cada distrito deba tener la misma población es la que se considera más importante; y por lo tanto, la propiedad a ser calculada es la población de cada polígono. Luego los polígonos se ordenan en orden creciente en base a esta propiedad calculada, y se eligen los primeros *k* en la lista ordenada que (1) que violen las restricciones de “*deben ir juntos*”, por ejemplo la contigüidad espacial y (2) cumplen las restricciones de “*no pueden ir juntos*”, donde *k* es la cantidad predefinida de conglomerados a detectar.

Una vez que se han seleccionado las semillas, los conglomerados iniciales se crean y puede comenzar el proceso de búsqueda. Para esto se adopta una búsqueda de tipo A\* en la que se asume que cada conglomerado inicial (que en el inicio está constituido simplemente por un solo polígono (la semilla)) es el estado inicial y que los conglomerados objetivo son el estado final. Consecuentemente, cada conglomerado crece a partir del estado inicial mediante el agregado de un polígono a la vez hasta obtener la conglomeración objetivo. El paradigma de búsqueda se adapta haciendo que al comienzo de cada iteración se busque el mejor conglomerado para hacer crecer (*MC*) y para esto se emplea la función heurística *(F)*, es asi que CPSC selecciona el conglomerado con la mayor necesidad, es decir el conglomerado con el mayor valor de *F*.

Una vez elegido el mejor conglomerado, el siguiente paso es elegir el mejor polígono (*MP*) para ser añadido al mismo. Para esto, primero se selecciona un conjunto de polígonos potenciales (*PP*) que pueden ser añadidos al *MC*. Este conjunto está constituido de todos los polígonos no asignados a un conglomerado y espacialmente contiguos al MC (es decir vecinos de *MC*, que es lo mismo que decir vecinos de los polígonos constituyentes de *MC*), que puede pensarse como polígonos que comparten su frontera con *MC* (*i.e.* con alguno de los polígonos constituyentes de *MC*). En el caso no existan polígonos no asignados libres dentro de la vecindad de *MC* entonces los polígonos vecinos que forman parte de conglomerados vecinos, donde cada uno de estos polígonos debe respetar las restricciones intra-conglomerado. Se selecciona entonces un polígono de este conjunto sobre la base de una heurística *F’*, que es una combinación de (1) una función *(H’)* que aproxima la distancia del estado actual del *MC* al estado objetivo una vez que se le agrega el *MP*, y (2) una función de costo *(G’)* que mide la reducción en la flexibilidad del crecimiento del *MP* luego de la anexión del *MP*. De esta forma CPSC selecciona el polígono que más contribuye al conglomerado, y por lo tanto *MP* es el polígono que resulta en el menor *F* para el *MC*. Esta estrategia de seleccionar el conglomerado con el mayor *F* como *MC* y luego elegir el polígono *MP* como el que resulta con el menor *F* para *MC*, permite que cada conglomerado crezca en forma simultánea y por lo tanto le da a cada uno la misma probabilidad de seleccionar por sí mismo el mejor polígono. Si por el contrario, *MC* se tomara como el conglomerado con el menor *F*, los conglomerados serian forzados a crecer en forma secuencial, y se perdería a propiedad de compacidad.

Luego de que se elige *MP* para ser añadido a *MC*, es necesario controlar que por anexión, la contigüidad espacial de los conglomerados se mantiene. Si *MC* y sus vecinos son espacialmente contiguos, *MP* se anexa al mejor conglomerado *MC* y el proceso continúa hasta que:

1. Todos los polígonos en el *dataset* han sido asignados a un conglomerado y se obtuvo el estado objetivo de conglomeración que satisface el conjunto de restricciones aplicadas, O
2. El algoritmo entra en un punto muerto (*deadlock*) donde los conglomerados entran en un ciclo repetitivo de estados.

La primer condición es fácil de entender, la segunda, implica que un conglomerado agrega un polígono, y luego lo pierde a manos de otro conglomerado, para luego volver a ganarlo, y volver a perderlo, y así sucesivamente[[1]](#footnote-1). Formalmente, es posible definir un conjunto de conglomerados que se encuentran en un “punto muerto” cuando en la iteración *I* un conglomerado *Cr* se encuentra en el estado *x*, y en la iteración *J*, donde *J ≤ I + k* el conglomerado *Cr* se encuentra nuevamente en el estado *x*. El estado de un conglomerado en cualquier iteración *I* se refiere a los polígonos que son sus miembros en dicha iteración. El algoritmo CPSC presentado por Joshi (XXXXXX tesis) puede ser aplicado a cualquier dominio, provisto que se tenga el conjunto de unidades base (polígonos), la cantidad de conglomerados a detectar, el conjunto de restricciones y la función heurística *F* que se basa en dichas restricciones:

**Algoritmo CPSC (Joshi XXXX Tesis)**

**Input:** Dataset D de n polígonos;

Función heurística F = G + H;

Cantidad de semillas k;

Estado objetivo (target) para los conglomerados;

Conjunto de restricciones intra-cluster.

1. **Elegir k semillas** {s1, s2, …, sk}
2. **Inicializar** k conglomerados asignando cada semilla un conglomerado (C1 = s1, C2 = s2, …, Ck = sk)
3. **While**(exista un polígono que no ha sido asignado a un conglomerado, O existe un conglomerado que no satisface todas las restricciones a nivel del conglomerado, O existe un estado de “punto muerto”)
   1. **Elegir el mejor conglomerado (MC) para crecer**
   2. **Encontrar la lista de polígonos posibles (PP)** como los candidatos para hacer crecer al MC.
   3. **Elegir el mejor polígono (MP)** a partir de PP para ser añadido a MC.
   4. **Agregar** MP a MC.
   5. **Actualizar el Estado de los Conglomerados.**
   6. **Si** se actualizo, continuar,

**Else** (“punto muerto” detectado), **Break.**

**End While.**

|  |  |
| --- | --- |
| **Elegir k semillas**  **Elegir** k semillas {s1, s2, …, sk} tales que:   1. La semilla debe ser un polígono con un mayor valor de F que los otros polígonos no semilla. 2. Cada polígono semilla debe violar las restricciones intra-cluster con relación a los otros polígonos semilla.   **Return** k semillas | **Elegir el mejor conglomerado (MC)**   1. **Calcular** F para cada conglomerado.   **Ver el Elegir del algoritmo CPSC\***  **Input:** Dataset D de n polígonos, F, k, Estado Objetivo para los conglomerados, Conjunto de restricciones intra conglomerado. |
| **Encontrar los polígonos posibles (PP)**  **Elegir** un conjunto de polígonos vecinos pp tales que pp = {p1, p2, …, pm}   1. El polígono pi cumple Contigüidad\_espacial(MC,pi) = true; 2. El polígono pi está libre (no ha sido asignado a un conglomerado; 3. El polígono pi no viola ninguna restricción intra-conglomerado;   **Si** pp = ∅, elegir un conjunto de polígonos vecinos pn, tales que:   1. El polígono pi cumple Contigüidad\_espacial(MC,pi) = true; 2. El polígono pi no viola ninguna restricción intra-conglomerado; 3. El polígono pi no fue añadido al MC en la iteración previa a la previa (iteración actual – 2);   **Return** pp | **Elegir el mejor polígono**  **Calcular** el resultado  **Do**  **Elegir** MP = pi|mini=0 to m(F’(MC+pi))  **IF** MP pertenece a un cluster vecino Cj  **IF** MP puede ser removido de Cj sin romper su contigüidad espacial; **return** true  **ELSE** **return** false  **IF** false,  **Hacer** MP = null;  **Remover** MP de PP;  **While**(MP = null)  **Return** MP |
| **Actualizar Estado del Conglomerado**  **IF** MP era un polígono libre, **Incrementar** iteración  **Return** true  **ELSE** **Incrementar** iteración  **Inicializar Monitor deadlock**  **IF** se detecta deadlock, **Return** false  **ELSE** **Return** true | **Monitor Deadlock**  **Agregar** MC e iteracion actual a la lista de monitoreo deadlock  **IF** dentro de los k – 1 items almacenados en la lista deadlock  **IF** las iteraciones almacenadas están en orden consecutivo  **IF** MC es un miembro de los k – 1 items  **Return** true  **Else**  **Return** false. |

**Extensiones al algoritmo CPSC**

A pesar que el algoritmo garantiza completitud y optimación en la solución, no garantiza convergencia dado que el proceso de búsqueda está centrado en el conglomerado en lugar de estar centrado en la instancia o polígono.

Joshi (XXXXX tesis) define convergencia de la siguiente manera (pág. 117):

“Un algoritmo se dice que converge cuando cada polígono pi ∈ *D*, donde *D* es el conjunto completo (dataset completo), es asignado a un conglomerado *C*j (*i.e.* .”

CPSC no garantiza convergencia, ya que todo polígono puede no ser asignado a un conglomerado. Ahora bien, *si el problema permite relajar las restricciones*, es posible garantizar convergencia con una versión modificada del algoritmo, que Joshi (XXXXX tesis) denomina CPSC\*.

CPSC\* sigue una estrategia similar a la de CPSC, pero *permite que el usuario relaje las restricciones para asegurar que todo polígono sea asignado a un conglomerado.* El proceso de relajamiento de restricciones ocurre en dos pasos. Primero, CPSC\* emplea una función de distancia ponderada *H* convirtiendo las restricciones duras a nivel del conglomerado en restricciones blandas y permitiendo al usuario priorizar las mismas. Segundo, mientras selecciona el polígono potencial para crecer un conglomerado, CPSC\* comprueba que todas las restricciones “deben ir juntos” se cumplan. Aun asi, en caso que el mejor conglomerado *MC* no haya alcanzado su estado objetivo, y no existan más polígonos disponibles que satisfagan las restricciones deseadas, entonces relaja las mismas de manera que los restantes polígonos puedan convertirse en miembros potenciales del *MC*.

La función ponderada de distancia *H* se define como:

(6)

donde es la distancia entre el estado actual del conglomerado *Cj* y su estado objetivo basándose en la restricción a nivel del conglomerado x, y *wx* es la ponderación asignada a esa restricción, donde *wx >* 0 y w1 + w2 + … + wx = 1. Las ponderaciones se asignan de acuerdo con la prioridad que el usuario define. De esta forma, la función ponderada de distancia *H* introduce flexibilidad en el crecimiento de los conglomerados en base a la elección de las restricciones y el peso que de las mismas en el problema. Por otro lado, para asegurar convergencia, CPSC\* inicia el monitoreo de “punto muerto” tan pronto como el conglomerado anexa un polígono que estaba asignado previamente a otro conglomerado. Este monitor almacena el estado actual del conglomerado. Si a lo largo de *k -1* iteraciones consecutivas, dos o más conglomerados repiten el mismo estado, se declara una situación de “punto muerto”. En este punto CPSC\* rompe el punto muerto empleando una estrategia que se detallara más abajo.

Es importante notar que conforme el **Teorema 1 (Joshi XXXX Tesis),** *CPSC\* garantiza convergencia* en el sentido definido anteriormente.

**Algoritmo CPSC\* (Joshi XXXX Tesis)**

**Input:** Dataset D de n polígonos;

Función heurística F = G + H;

Cantidad de semillas k;

Estado objetivo (target) para los conglomerados;

Conjunto de restricciones intra-cluster.

1. **Elegir k semillas** {s1, s2, …, sk}
2. **Inicializar** k conglomerados asignando cada semilla un conglomerado (C1 = s1, C2 = s2, …, Ck = sk);

**Inicializar** iteración = 0;

1. **While**(exista un polígono que no ha sido asignado a un conglomerado)
   * 1. **Elegir el mejor conglomerado (MC) para crecer**
     2. **Encontrar la lista de polígonos posibles (PP)** para ser agregados a MC.
     3. **Elegir el mejor polígono (MP)** de PP para ser añadido a MC.
     4. **Actualizar el Estado de los Conglomerados**
     5. **Si** se actualizo, Continuar,

**Else** (“punto muerto” detectado)

**Elegir** el MC entre los conglomerados que no participan del “punto muerto”

Repetir el proceso de crecimiento para MC

**End While.**

|  |  |
| --- | --- |
| **Encontrar los polígonos posibles (PP)**  **Elegir** un conjunto de polígonos vecinos pn = {p1, p2, …, pm} tales que:   * El polígono pi cumple Contigüidad\_espacial(MC,pi) = true donde Contigüidad\_espacial es la función de contigüidad que se dio como input al algortimo; * El polígono pi está libre (no ha sido asignado a un conglomerado; * El polígono pi no viola ninguna restricción intra-conglomerado;   **IF** pn = ∅, **then** elegir un conjunto de polígonos vecinos pn = {p1, p2, …, pn}, tales que:   * El polígono pi cumple Contigüidad\_espacial(MC,pi) = true y Contigüidad\_espacial es la función de contigüidad que se dio como input al algortimo; * El polígono pi no viola ninguna restricción intra-conglomerado;   **Si** pn = ∅, **then** relajar las restricciones intra-cluster y repetir el proceso de selección de polígonos posibles;  **Return** pn | **Elegir el mejor conglomerado (MC)**  **Calcular** F para cada conglomerado.  **Elegir** el mejor conglomerado , es decir, el conglomerado con el mayor F.  **IF** dos o mas conglomerados (Cj, Ci) tienen el mayor F, es decir F(Cj) = F(Ci) y Cj ≠ Ci  **Elegir** el mejor conglomerado Cj tal que:  N\_poligonos\_libres(Cj) > 0 y  N\_poligonos\_libres(Cj) < n\_poligonos\_libres(Ci)  donde n\_poligonos\_libres(Cj) es la cantidad de polígonos libres espacialmente contiguos a Cj.  **Return** el mejor conglomerado Cj. |
| **Elegir k semillas**  **Elegir** k semillas {s1, s2, …, sk} tales que:   * La semilla debe ser un polígono con un mayor valor de F que los otros polígonos no semilla; * Cada semilla debe ser no contigua con cada otra semilla basándose en la función de contigüidad elegida;   **Return** k semillas | **Elegir el mejor polígono (MP)**  **Calcular** el resultado F’  **Elegir** MP = pi|min0≤i≤q F’(MC+pi))  **IF** MP pertenece a un cluster vecino Cj  **IF** MP puede ser removido de Cj sin romper su contigüidad espacial; **return** true  **ELSE** **return** false  **IF** true, **then** MP permanence el mismo;  **ELSE**, remover MP de PP;  *Repetir el proceso de seleccion de MP*  **Return** MP |
| **Actualizar Estado del Conglomerado**  **Agregar** MP a MC;  **IF** MP era un polígono libre, **Incrementar** iteración  **Return** true  **ELSE** **Incrementar** iteración  **Inicializar Monitor deadlock**  **IF** se detecta deadlock, **Return** false  **ELSE** **Return** true | **Monitor Deadlock**  **Agregar** MC e iteración actual a la lista de monitoreo deadlock  **IF** dentro de los k – 1 previos items almacenados en la lista deadlock  **IF** las iteraciones almacenadas están en orden consecutivo  **IF** MC es un miembro de los k – 1 items  **Return** true  **Else**  **Return** false. |

**Aplicación de CPSC al distritado electoral**

Es posible plantear el problema de distritado como uno de formación de conglomerados donde cada conglomerado representa un distrito (sección electoral / circuito, etc.). Cada distrito o conglomerado se forma agrupando polígonos que siguen ciertas restricciones.

Para distritado electoral Joshie emplea CPSC puro.

En este caso, es posible definir las restricciones principales como:

1. Todos los distritos deben tener la misma población;
2. Cada distrito debe definir un territorio contiguo;
3. Los distritos deben ser compactos;
4. Los distritos deben respetar las comunidades de interés;
5. Los distritos deben conformar las fronteras naturales y antrópicas existentes, en la medida de lo posible;

En caso de conflicto, generalmente se da mayor importancia a la igualdad de población y la contigüidad espacial; siguiendo a Joshi (XXXXX tesis), nuestra implementación respetara las tres primeras restricciones. Las demás restricciones pueden incorporarse como restricciones a nivel de la instancia en el algoritmo CPSC.

El problema queda expresado entonces como:

*Dividir el área en estudio en k distritos tales que la población total dentro de cada distrito sea aproximadamente la misma o dentro de un α % de error. Cada uno de los k distritos debe ser espacialmente contiguo. Finalmente todos los k distritos deben ser tan compactos como sea posible.*

**Heurística empleada:** La función *F* empleada por CPSC para determinar el mejor conglomerado a crecer y el mejor poligono a anadir se define basándose en el dataset de entrada y las restricciones definidas precedentemente antes del proceso de conglomeracion. Los inputs para el algoritmo son:

**Dataset:** Conjunto de partidos de la provincia de Buenos Aires (Para el caso de las secciones electorales)

Conjnto de radios censales del partido de La Plata (Para los circuitos electorales).

**Cantidad de Semillas**: k

**Target (Objetivo):** Conglomerados espacialmente contiguos y compactos {C1, C2, …, Ck} cada uno conteniendo una población *x* dentro de un margen de error del α %.

**Conjunto de Restricciones:** *Restricciones a nivel del Conglomerado*

CS1: Cada conglomerado debe ser espacialmente contiguo.

CS2: Cada conglomerado debe ser compacto.

1. Notemos en este punto la analogía con la búsqueda tabú, donde justamente la memoria tabú se diseña para evitar caer en estos ciclos repetitivos. [↑](#footnote-ref-1)