Morfismos, Vol. 18, No. 1, 2014, pp. 45–55

Captura de objetos móviles sobre una recta *

Luis E. Urbán Rivero Rafael López Bracho
Francisco J. Zaragoza Martínez

Resumen

En el problema del agente viajero euclidiano se tiene un conjunto de n puntos en el plano y se desea que un agente los visite todos recorriendo la mínima distancia euclidiana posible. Presentamos una variante de este problema en la que los puntos son móviles y existen durante un tiempo finito sobre una recta fija. Usando técnicas de programación lineal encontramos algoritmos de tiempo polinomial, que verifican si un orden de captura dado es factible y, en ese caso, minimizan el tiempo de captura y la distancia total recorrida.

2010 Mathematics Subject Classification: 90C08, 90C90.
Keywords and phrases: Problema del agente viajero, objetos móviles.

1. Introducción

En el problema del agente viajero (Traveling Salesman Problem, TSP) se tiene un conjunto de n ciudades y se pretende encontrar un recorrido de costo mínimo para visitar todas las ciudades y regresar al lugar donde empezó el recorrido. Si pensamos en la versión euclidiana de este problema en dos dimensiones (Euclidean TSP), cada lugar a visitar es un punto en el plano euclidiano y el costo de moverse de un punto a otro es la distancia euclidiana. Se sabe que TSP es NP completo [5] y que Euclidean TSP es NP duro [3].

*Este trabajo es parte de la tesis de maestría del primer autor que será presentada en el Posgrado de Optimización de la UAM Azcapotzalco bajo la supervisión del Dr. Rafael López Bracho y el Dr. Francisco Javier Zaragoza Martínez, ambos del Departamento de Sistemas de la UAM Azcapotzalco.

Partiendo de lo anterior, podemos imaginar que cada lugar a visitar es un punto móvil en el plano que se desplaza con su propia velocidad constante, como se muestra en la Figura 1. Ésta es una generalización de Euclidean TSP, puesto que en él todos los puntos tienen una velocidad de 0. Se ha demostrado que si la cantidad de objetos que se mueven es a lo más $O(\log n / \log \log n)$, se tiene una garantía aproximación de $1 + \alpha$, donde α es la garantía para un algoritmo de aproximación de Euclidean TSP. A este problema se le conoce como agente viajero con objetivos móviles (Moving Target TSP, MTTSP) [1]. Por lo anterior, podemos decir que MTTSP es al menos tan complejo como Euclidean TSP.

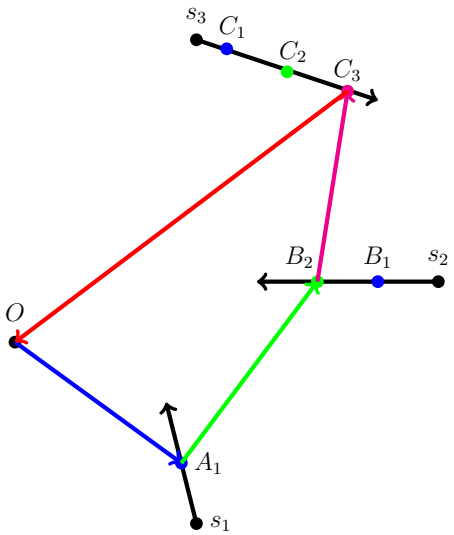

Figura 1: Ejemplo de MTTSP donde existen tres objetos móviles s_1, s_2 y s_3. También se muestran las posiciones de captura (A_1, B_2, C_3) de cada uno de ellos, dado su vector movimiento.

2. Problemas de captura sobre una recta

En [1] se propone una variante de MTTSP donde los puntos móviles a capturar están sobre una misma línea recta, se comienza en el origen y no se tiene que regresar a ese mismo punto. Para esa variante se muestra un algoritmo de orden cuadrático que lo resuelve. Es a partir de esta variante como llegamos a los problemas que plantea este artículo, en donde la diferencia principal radica en que los objetos tienen un intervalo

de existencia sobre la recta y una posición de aparición, como se muestra en la Figura 2.

El primer problema, que llamaremos captura de todos los objetos móviles sobre una recta (CTOMSR), se define como sigue: Dado un agente con rapidez variable (acotada superiormente por una constante U) y n objetos móviles, cada uno con un momento de aparición (a_i), un momento de desaparición (d_i), una posición de aparición (p_i) y una velocidad (v_i), se quiere decidir si el agente es capaz de capturar todos los objetos en el orden $1, 2, \ldots, n$. En caso de que sí se pueda, se pueden formular otros dos problemas: por un lado se desea minimizar el tiempo de captura (CTOMSR rápido) y por otro lado se desea minimizar la distancia total recorrida (CTOMSR perezoso).

En estos problemas dar un orden predefinido de captura tiene sentido, porque en principio se tiene que determinar si es posible capturar a todos los objetos, situación que marca una diferencia respecto al Euclidean TSP donde ello resulta ser trivial. Se sabe que el problema de determinar si es posible capturar a todos los objetos sin un orden predefinido de captura es NP Completo [4]. Por otro lado, si además no se tuvieran intervalos de tiempo se sabe que el problema se puede resolver en tiempo polinomial [1]. Esta situación intermedia determina la importancia de los tres problemas a tratar.

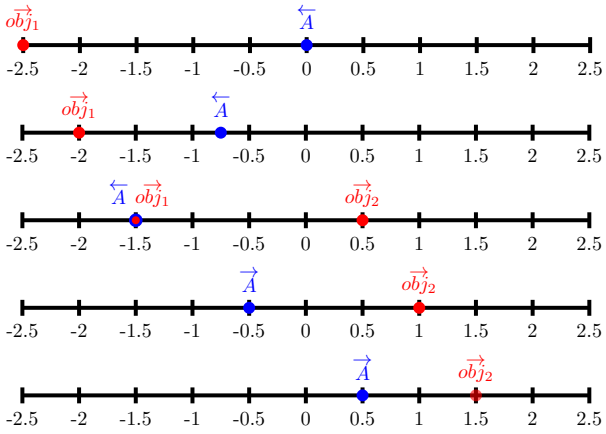

Figura 2: Ejemplo del problema propuesto que muestra cómo se mueve el agente con rapidez entre 0 y 1 en la recta, a la izquierda con rapidez 0.75 y a la derecha con rapidez 1 (en distintos periodos de tiempo), así como la aparición y desaparición de objetos móviles.

2.1. Modelos lineales

Ahora podemos ubicar la posición y el tiempo en un plano, con lo cual los puntos móviles sobre la recta se convertirán en segmentos de recta en el plano posición vs tiempo (x vs t).

Si x_i y t_i son la posición y el momento de captura del objeto i, es necesario que se capture en el intervalo de existencia del objeto, es decir:

$$a_i \leq t_i \leq d_i.$$

También se necesita que dicho punto esté sobre la recta que describe el segmento que queremos intersectar, es decir:

$$x_i - p_i = v_i(t_i - a_i).$$

Además, el agente debe poder capturar al objeto o, dicho de otro modo, el objeto debe estar en el rango de intercepción del agente, es decir:

$$-U(t_i - t_{i-1}) \leq x_i - x_{i-1} \leq U(t_i - t_{i-1}).$$

Adicionalmente se necesita la relación de orden de captura, es decir:

$$t_{i-1} \leq t_i.$$

El modelo descrito se puede visualizar en la Figura 3.

Aunque la posición inicial del agente pudiera ubicarse en cualquier punto de la recta, sin pérdida de generalidad supondremos que el agente partirá del origen, es decir $t_0 = 0$ y $x_0 = 0$. Por último, sólo nos queda decir que el tiempo es no negativo $t_i \geq 0$ y la posición x_i es libre porque se puede capturar cualquier objeto en cualquier parte de la recta donde el objeto exista.

El modelo lineal completo del problema de capturar todos los objetos móviles sobre una recta (CTOMSR) queda como sigue:

$$
\begin{aligned}
(1) \quad & t_i \geq a_i \quad i = 1, \ldots, n \\
(2) \quad & t_i \leq d_i \quad i = 1, \ldots, n \\
(3) \quad & x_i - p_i = v_i(t_i - a_i) \quad i = 1, \ldots, n \\
(4) \quad & x_i - x_{i-1} \geq -U(t_i - t_{i-1}) \quad i = 1, \ldots, n \\
(5) \quad & x_i - x_{i-1} \leq U(t_i - t_{i-1}) \quad i = 1, \ldots, n \\
(6) \quad & t_{i-1} \leq t_i \quad i = 1, \ldots, n \\
(7) \quad & t_0 = 0 \\
(8) \quad & x_0 = 0 \\
(9) \quad & t_i \geq 0 \quad i = 1, \ldots, n \\
(10) \quad & x_i \text{ libre} \quad i = 1, \ldots, n
\end{aligned}
$$

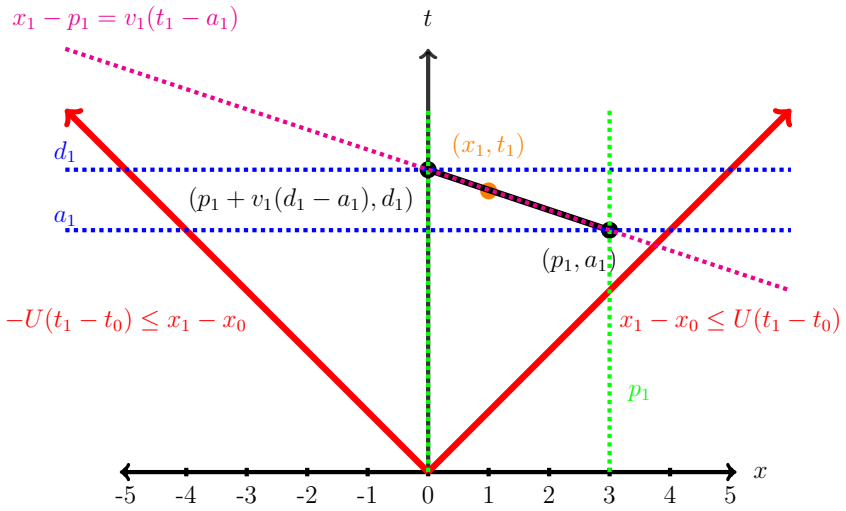

Figura 3: Diagrama x vs t con todas las restricciones del objeto 1.

Observe que la restricción (6) se puede quitar del modelo puesto que queda implícita en la restricciones (4) y (5).

Para el caso del CTOMSR rápido, es posible despejar la variable x_i en la igualdad (3) del modelo anterior, el nuevo modelo quedaría en términos de t_i:

$$\text{(11)} \qquad \text{mín}\, z = t_n$$
$$\text{(12)} \qquad t_i \geq a_i \quad i = 1, \ldots, n$$
$$\text{(13)} \qquad -t_i \geq -d_i \quad i = 1, \ldots, n$$
$$\text{(14)} \qquad (U - v_i)t_i - (U + v_{i-1})t_{i-1} \geq \alpha_i \quad i = 1, \ldots, n$$
$$\text{(15)} \qquad (U + v_i)t_i - (U - v_{i-1})t_{i-1} \geq -\alpha_i \quad i = 1, \ldots, n$$
$$\text{(16)} \qquad t_0 = 0$$
$$\text{(17)} \qquad t_i \geq 0 \quad i = 1, \ldots, n$$

donde las $\alpha_i = a_i v_i - a_{i-1} v_{i-1} - p_i + p_{i-1}$ son constantes.

Para el caso del CTOMSR perezoso, el modelo ahora requiere del despeje de las t_i y queda en términos de x_i (suponiendo sin pérdida de generalidad que ninguna $v_i = 0$).

$$\text{mín } z = \sum_{i=1}^{n} r_i \tag{18}$$

$$\frac{x_i}{v_i} \geq \frac{p_i}{v_i} \quad i = 1, \ldots, n \tag{19}$$

$$-\frac{x_i}{v_i} \geq a_i - d_i - \frac{p_i}{v_i} \quad i = 1, \ldots, n \tag{20}$$

$$-x_i + x_{i-1} + r_i \geq 0 \quad i = 1, \ldots, n \tag{21}$$

$$x_i - x_{i-1} + r_i \geq 0 \quad i = 1, \ldots, n \tag{22}$$

$$\left(\frac{U}{v_i} + 1\right) x_i - \left(\frac{U}{v_{i-1}} + 1\right) x_{i-1} \geq \delta_i \quad i = 1, \ldots, n \tag{23}$$

$$\left(\frac{U}{v_i} - 1\right) x_i - \left(\frac{U}{v_{i-1}} - 1\right) x_{i-1} \geq \delta_i \quad i = 1, \ldots, n \tag{24}$$

$$x_0 = 0 \tag{25}$$

$$r_i \geq 0 \quad i = 1, \ldots, n \tag{26}$$

$$x_i \quad \text{libre} \quad i = 1, \ldots, n \tag{27}$$

en donde las r_i son variables auxiliares para eliminar un valor absoluto de la función objetivo que era originalmente $\sum_{i=1}^{n} |x_i - x_{i-1}|$ y convertirla en (18) y con

$$\delta_i = U \left(a_{i-1} - \frac{p_{i-1}}{v_{i-1}} - a_i + \frac{p_i}{v_i} \right)$$

constante.

Note que si alguna $v_i = 0$, entonces $x_i = p_i$ y el modelo se simplificaría.

3. Algoritmos

Con los modelos anteriores y los siguientes resultados podemos decir que tanto el problema de la factibilidad como el de minimización de tiempo y distancia total recorrida se pueden resolver en tiempo polinomial partiendo del algoritmo de Karmarkar [2].

Teorema 3.1 (Karmarkar)**.** *La complejidad del algoritmo de Karmarkar para un programa lineal de la forma* mín cx *sujeto a* $Ax \geq b$ *con* p

variables y q restricciones es de

$$O(p^{3.5}(|A| + |b| + |c|)^2),$$

donde $|A|$, $|b|$ y $|c|$ son la cantidad de bits necesarios para codificar la matriz A, b y c respectivamente.

Corolario 3.2. *Para el problema* ***CTOMSR rápido*** *existe un algoritmo basado en el algoritmo de Karmarkar y se ejecuta en tiempo*

$$O(\ell^2 n^{5.5})$$

donde ℓ es la máxima cantidad de bits que se requieren para almacenar U y cualquier a_i, d_i, p_i o v_i.

Demostración. Con el modelo de minimización del tiempo de captura (11–17) podemos obtener los siguientes tamaños:

$$|A| = (8\ell + 2)n, \quad |b| = 6\ell n, \quad |c| = 1.$$

El número de operaciones requeridas se puede obtener mediante el Teorema 3.1 y es $O(n^{3.5}(14\ell n + 2n + 1)^2)$, lo cual está en $O(\ell^2 n^{5.5})$. □

Corolario 3.3. *Para el problema* ***CTOMSR perezoso*** *existe un algoritmo basado en el algoritmo de Karmarkar y se ejecuta en tiempo*

$$O(\ell^2 n^{5.5})$$

donde ℓ es la máxima cantidad de bits que se requieren para almacenar U y cualquier a_i, d_i, p_i o v_i.

Demostración. Con el modelo de minimización de la distancia total recorrida (18–27) podemos obtener los siguientes tamaños:

$$|A| = (10\ell + 6)n, \quad |b| = 8n\ell, \quad |c| = n.$$

El número de operaciones requeridas se puede obtener mediante el Teorema 3.1 y es $O(n^{3.5}(18n\ell + 7n)^2)$ lo cual está en $O(\ell^2 n^{5.5})$. □

3.1. Algoritmo lineal para CTOMSR rápido

A continuación presentaremos un algoritmo que resuelve CTOMSR rápido en tiempo lineal. Para ello resolveremos el problema por etapas. Comenzando en el origen, se determina el primer (t_1^e) y el último

momento (t_1^f) en que se puede capturar el primer objeto. Con esa información se calcula lo mismo para el segundo objeto y así sucesivamente hasta calcular t_n^e y t_n^f. Es decir, en cada etapa se resuelven dos programas lineales con funciones objetivo $t_i^e = \text{mín}(t_i)$ y $t_i^f = \text{máx}(t_i)$ y el conjunto de restricciones (28–34), para las variables t_i y t_{i-1}:

$$t_i \geq a_i \tag{28}$$

$$-t_i \geq -d_i \tag{29}$$

$$t_{i-1} \geq t_{i-1}^e \tag{30}$$

$$-t_{i-1} \geq -t_{i-1}^f \tag{31}$$

$$(U - v_i)t_i - (U + v_{i-1})t_{i-1} \geq \alpha_i \tag{32}$$

$$(U + v_i)t_i - (U - v_{i-1})t_{i-1} \geq -\alpha_i \tag{33}$$

$$t_i \geq 0 \tag{34}$$

donde $\alpha_i = a_i v_i - a_{i-1}v_{i-1} - p_i + p_{i-1}$ son constantes, además t_i^e y t_i^e se calculan en el paso anterior, con la excepción de $t_0^e = 0$ y $t_0^f = 0$. Estos programas lineales se deducen del programa lineal para CTOMSR rápido (11–17).

En este momento tenemos el tiempo mínimo y el tiempo máximo para capturar a todos los objetos. Si además se desea encontrar una ruta factible para dicho tiempo mínimo, se toma $t_n'^e = t_n^e$, $t_n'^f = t_n^e$ y se resuelven dos programas lineales con funciones objetivo $t_{i-1}'^e = \text{mín}(t_{i-1})$ y $t_{i-1}'^f = \text{máx}(t_{i-1})$ y el conjunto de restricciones (28–34), para las variables t_i y t_{i-1}, modificando las dos primeras restricciones como sigue:

$$t_i \geq t_i'^e \tag{35}$$

$$-t_i \geq -t_i'^f \tag{36}$$

donde $t_i'^e$ y $t_i'^f$ se calculan en el paso anterior.

Adicionalmente se buscará la ruta que ocupará el tiempo más tardío mediante un procedimiento similar pero con $t_n''^e = t_n^f$, $t_n''^f = t_n^f$ y el conjunto de restricciones (28–34), para las variables t_i y t_{i-1}, modificando las dos primeras restricciones como sigue:

$$t_i \geq t_i''^e \tag{37}$$

$$-t_i \geq -t_i''^f \tag{38}$$

donde $t_i''^e$ y $t_i''^f$ se calculan en el paso anterior.

Algoritmo por etapas

1. Sea $t_0^e = 0$, $t_0^f = 0$.
2. Para $i = 1, \ldots, n$:
   a) Resolver los programas lineales $t_i^e = \text{mín}(t_i)$ sujeto a (28–34) y $t_i^f = \text{máx}(t_i)$ sujeto a (28–34).
3. Sea $t_n'^e = t_n^e$, $t_n'^f = t_n^e$.
4. Para $i = n, \ldots, 1$:
   a) Resolver los programas lineales $t_{i-1}'^e = \text{mín}(t_{i-1})$ sujeto a (35–36) y (30–34) y $t_{i-1}'^f = \text{máx}(t_{i-1})$ sujeto a (35–36) y (30–34).
5. Sea $t_n''^e = t_n^f$, $t_n''^f = t_n^f$.
6. Para $i = n, \ldots, 1$:
   a) Resolver los programas lineales $t_{i-1}''^e = \text{mín}(t_{i-1})$ sujeto a (37–38) y (30–34) y $t_{i-1}''^f = \text{máx}(t_{i-1})$ sujeto a (37–38) y (30–34).

Teorema 3.1.1. *El problema* ***CTOMSR rápido*** *es resuelto en tiempo lineal por el algoritmo por etapas.*

Demostración. Cada uno de los programas lineales que se deben resolver en el paso 2a del algoritmo consta de 2 variables y 6 desigualdades. Se sabe por la teoría de la programación lineal que la solución óptima se puede obtener resolviendo a lo más $\binom{2+6}{6} = 28$ sistemas de 6 ecuaciones lineales con 6 variables. Como cada uno de estos sistemas se puede resolver en tiempo constante, entonces el paso 2a se puede llevar a cabo también en tiempo constante. Es decir, el paso 2 del algoritmo se puede completar en tiempo lineal. Evidentemente, lo mismo es cierto para los pasos 4 y 6 del algoritmo. Finalmente, como los pasos 1, 3 y 5 son de tiempo constante, el tiempo de ejecución del algoritmo es lineal. □

En la Figura 4 se muestra una salida del algoritmo por etapas. Los segmentos de línea continua son los objetos móviles. La zona de líneas verticales y contorno punteado corresponde con la región a través de la cual cualquier ruta requiere de tiempo mínimo t_n^e para capturar a todos los objetos. La zona de líneas horizontales y contorno rayado corresponde con la región a través de la cual cualquier ruta requiere de tiempo máximo t_n^f para capturar a todos los objetos. La región factible

está determinada por la región de tiempo máximo, la región de tiempo mínimo y la región sin ningún patrón entre estas dos.

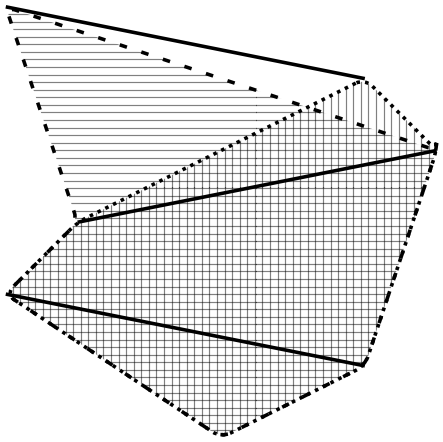

Figura 4: Ejemplo de la salida del algoritmo por etapas.

4. Conclusiones y trabajo futuro

En este trabajo se da una cota superior a la complejidad de los problemas CTOMSR, CTOMSR rápido y CTOMSR perezoso. Para los dos primeros casos se tiene un algoritmo de tiempo lineal, que asintóticamente no puede mejorarse más, y para el último caso se tiene un algoritmo polinomial basado en el algoritmo de Karmarkar. Es probable que se pueda encontrar un algoritmo que resuelva CTOMSR perezoso asintóticamente más rápido. Por otro lado, si no se pudiera capturar todos los objetos, entonces nos interesaría dar un algoritmo que maximice la cantidad de objetos atrapados (máx COMSR). Finalmente nos interesa estudiar el caso en que el orden de captura debe ser determinado por el algoritmo, para minimizar el tiempo de captura, minimizar la distancia total recorrida o maximizar la cantidad de objetos atrapados.

5. Agradecimientos

El primer autor recibió una beca del Acuerdo 02/2011 de la UAM y actualmente recibe una beca por parte del Conacyt en el Programa Nacional de Posgrados de Calidad. Los otros dos autores reciben apoyo

de la División de CBI de la UAM Azcapotzalco a través del proyecto *Algoritmos y modelos para problemas de optimización en redes*. El tercer autor recibe apoyo del Conacyt (SNI 33694). Los tres autores agradecen el tiempo dedicado, las valiosas sugerencias y comentarios del editor y los árbitros en el proceso de revisión.

Luis Eduardo Urbán Rivero
Posgrado en Optimizacion,
Universidad Autónoma Metropolitana Azcapotzalco,
Av. San Pablo 180
Col. Reynosa-Tamaulipas
Delegación Azcapotzalco
C.P. 02200 México, D.F.
al2122800366@alumnos.azc.uam.mx

Rafael López Bracho
Departamento de Sistemas,
Universidad Autónoma Metropolitana Azcapotzalco,
Av. San Pablo 180
Col. Reynosa-Tamaulipas
Delegación Azcapotzalco
C.P. 02200 México, D.F.
rlb@correo.azc.uam.mx

Francisco Javier Zaragoza Martínez
Departamento de Sistemas,
Universidad Autónoma Metropolitana Azcapotzalco,
Av. San Pablo 180
Col. Reynosa-Tamaulipas
Delegación Azcapotzalco
C.P. 02200 México, D.F.
franz@correo.azc.uam.mx

Referencias

[1] Helving C. S., Robins G., Zelikovsky A., The moving-target traveling salesman problem, Journal of Algorithms **49** (2003), 153–174.

[2] Karmarkar N., A new polynomial-time algorithm for linear programming, Combinatorica **4** (1984), 373–395.

[3] Papadimitriou C. H., The Euclidean travelling salesman problem is NP-complete, Theoretical Computer Science **4** (1977), 237–244.

[4] Tsitsiklis J. N., Special Cases of Traveling Salesman and Repairman Problems with Time Windows, Networks **22** (1992), 263–282.

[5] Karp R.M., Reducibility Among Combinatorial Problems, Complexity of Computer Computations, In R.E. Miller and J.W. Thatcher (1972), 85–103.