Anales PANEL'81/12 JAIIO
Sociedad Argentina de informática
e Investigación Operativa. Buenos Aires, 1981

ANALISIS DEL ALGORITMO DE KHACHIYAN Y ESTUDIO DE SU IMPLEMENTACION EN COMPUTADORA USANDO LENGUAJE BASIC

Eleonora Poyard de Vincenti

Sector Computación y Cálculo
Instituto Nacional de Tecnología Industrial
Buenos Aires - Argentina.

1. INTRODUCCION

A fines de 1979, a través de los medios de comunicación masiva de todo el mundo, surge la noticia de un nuevo método para la resolución de problemas de Programación Lineal que revoluciona al mundo de las Matemáticas. Su autor, el matemático ruso Leonid Khachiyan, se hace famoso de la noche a la mañana.

Cuál es el extraordinario logro de Khachiyan?

Las versiones de los distintos diarios y publicaciones son muy variadas e incluso contradictorias, lo cual presenta un desafío a todos los especialistas en el tema de Programación Lineal, abriendo las puertas a la investigación y a la experimentación. Es que, con el nuevo algoritmo, aumenta la esperanza de superar el mayor impedimento del uso del clásico método SIMPLEX. Nos referimos, por supuesto, a la imposibilidad de determinar -a priori- el tiempo aproximado que llevará resolver un problema dado aplicando el SIMPLEX, método que, por otra parte, no ofrece ninguna duda con respecto a su eficacia.

Es en este punto, precisamente, donde radica lo revolucionario del nuevo método: se conoce, desde el primer momento, una cota superior para el número de pasos necesarios para resolver un problema dado y se puede asegurar, sin ninguna duda, que si la cantidad de iteraciones supera este límite sin encontrarse una solución, no existe ninguna solución para el problema. Llamando N a la cota superior mencionada, resulta $N = 16n^2L$, donde n es la cantidad de variables del problema y L está dado por

$$L = \left[\sum_{i=1}^{m} \sum_{j=1}^{n} \log_2 (a_{ij} + 1) + \sum_{i=1}^{m} \log_2 (b_i + 1) + \log_2 n.m\right] + 1$$

donde:

a_{ij} (i=1 m; j=1 n) son los coeficientes de la matriz de restricciones

b_i (i=1,m) son las coordenadas del vector de requerimientos y

m es la cantidad de inecuaciones del sistema de restricciones.

De esta manera, la cantidad de pasos necesarios para resolver un problema crece polinomialmente con el número de variables mientras que en el método SIMPLEX no se ha podido determinar aún cómo es el crecimiento aunque las únicas cotas superiores halladas (y comprobadas) hasta el momnto indican un crecimiento exponencial.

Estas asombrosas ventajas teóricas del método de Khachiyan encuentran su contrapartida en una (gran) desventaja en el orden práctico. Esta es el enorme nivel de precisión requerido para todos los cálculos: los valores involucrados en cada paso necesitan una precisión de 2^{-37nL}.

Como indican los números, a medida que el problema se hace más grande se reduce notablemente (respecto de otros métodos) el tiempo de resolución pero también se incrementa (exponencialmente) la precisión requerida por los cálculos.

2. ALGORITMO

Dado un sistema de m inecuaciones lineales con n incógnitas el algoritmo decide en primer lugar, si es consistente. Si no lo es la inconsistencia quedará probada si se producen $N=16n^2L$ iteraciones sin arrojar una solución, En caso contrario, el algoritmo encontrará un valor "x" que satisfaga las m inecuaciones en menos de N iteraciones.

Pára el caso de un problema de Programación Lineal de la forma:

$$\text{Maximizar} \quad Z = c^t . x$$

$$\text{sujeto a} \quad A.x \leq b \qquad (1)$$

$$\text{con} \quad x \geq 0$$

donde: $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{mxn}$

y cuyo problema dual es:

$$\text{Minimizar} \quad Z = b^t . y$$

$$\text{sujeto a} \quad A^t y \geq c \qquad (2)$$

$$\text{con} \quad y \geq 0$$

donde: $y \in \mathbb{R}^m$,

el algoritmo encuentra simultáneamente una solución para (1) y una solución para (2), transformando ambos problemas en un solo problema de resolución de un sistema de 2(m+n+1) inecuaciones lineales con m+n incógnitas. Este sistema ampliado se construye de la siguiente forma:

Sean x una solución de (1) e y una solución de (2). Debe verificarse:

$$c^t x \leq (A^t y)^t x = y^t Ax \leq y^t b = b^t y \Longrightarrow z \leq Z$$

Por lo tanto, cuando x e y son soluciones óptimas de (1) y (2) respectivamente, debe ser z = Z.

Es decir, se verifican simultáneamente las dos inecuaciones:

$$c^t x - b^t y \leq 0 \qquad y \qquad c^t x - b^t y \geq 0$$

Podemos agregar estas nuevas inecuaciones a las que ya teníamos obteniendo, como planteo general del problema a resolver:

Encontrar $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ que verifiquen:

$$c^t x - b^t y \leq 0$$

$$-c^t x + b^t y \leq 0$$

$$Ax \leq b$$

$$-A^t y \leq -c$$

$$-I_{nx} \leq 0$$

$$-I_{my} \leq 0$$

Encarado de esta manera, la resolución de un problema de Programación Lineal (y su dual) se reduce a la resolución de un sistema de inecuaciones. Si este sistema ampliado resulta consistente, encontrar una solución significa haber encontrado soluciones óptimas para los problemas primal y dual respectivamente.

2.1. Concepto Geométrico

Desde el punto de vista geométrico, el algoritmo pretende generar incialmente un elipsoide n-dimensional de volumen suficientemente grande como para encerrar el conjunto de soluciones del sistema y luego desplazar el mencionado elipsoide achicándolo alrededor del conjunto en forma tal de lograr, en el último paso, el elipsoide de mínimo volumen que contenga al conjunto de soluciones.

Como elipsoide inicial se toma una esfera centrada en el origen de radio 2^L (en el inciso 4. se explica por qué esta esfera es "suficientemente" grande). Las soluciones intentadas serán, en todos los pasos, los centros de las correspondientes elipses. Por lo tanto, el primer intento de solución será el 0. Se mide entonces cuánto dista el 0 de ser una solución registrando el número i de la inecuación más "violada" del sistema por este intento de solución. Luego se traslada el centro del elipsoide en la dirección determinada por i y se construye un elipsoide de menor volumen que siga conteniendo al conjunto de soluciones.

Idealmente (en la práctica es imposible) se busca que el nuevo elipsoide sea tangente al anterior en algún punto. Al iterar estas construcciones deberá llegar un momento en que el centro del elipsoide pertenezca al conjunto de soluciones y por consiguiente hayamos encontrado una solución explícita para nuestro problema. En el caso de un problema de Programación Lineal, la solución también será óptima.

Tal vez una de las características más importantes del algoritmo es que, después de $N = 16n^2L$ iteraciones hemos achicado el volumen a cero, con lo cual queda irrefutablemente comprobado que no existe un conjunto de soluciones y por lo tanto nuestro sistema es inconsistente.

Los requerimientos de volumen mínimo del elipsoide y tangencia con el inmediato anterior son factores demasiado exigentes con respecto a la precisión necesaria para los cálculos efectivos. A tal punto que, en la práctica, sería imposible lograr una solución.

Khachiyan intenta superar esta dificultad determinando volúmenes lo más grandes posible para las elipsoides pero tratando de no caer en tamaños desmedidos que impedirían achicarlos hacia el conjunto de soluciones con suficiente velocidad.

2.2 Descripción del algoritmo

En cada paso, correspondiente a la k-ésima iteración del algoritmo, se determinan:

x_k: el punto de $\mathbb{R}^n$ que se intenta como solución después de K iteraciones.

Q_k: una matriz simétrica definida positiva en $\mathbb{R}^{nxm}$ que representa un elipsoide E_k centrado en x_k, que contiene las soluciones del sistema, donde:

$$E_k = \left\{ y \in \mathbb{R}^n : y = x_k + Q_k \cdot z ; \; z \in \mathbb{R}^n , \| Z \| \leq 1 \right\}$$

$\theta_k(x_k)$: mide cuánto dista x_k de ser una solución.

Θ_k: registra el mejor intento de solución logrado hasta el momento al ser igual al menor de los previos $\theta_k(x_k)$

Inicializamos, para k=0, de la siguiente forma:

$$x_o = 0$$

$$Q_o = 2^L x I_n$$

$$\theta_o = \theta_o(x_o) = \underset{1=i=m}{\text{máx}} (-b_i)$$

Esto equivale a tomar como primer intento de solución al 0, y como primer elipsoide a la esfera de radio 2^L centrada en el origen, la cual tiene suficiente volumen como para contener a todos los vértices del conjunto convexo de soluciones.

Una vez obtenidos los valores correspondientes al paso k, reemplazamos el x_k obtenido en el sistema de inecuaciones y, comparando con los respectivos coeficientes del vector de restricciones, calculamos la discrepancia actual:

$$\theta_k(x_k) = \text{máx} (a_i^t \cdot x_k - b_i)$$

Guardamos el valor i(k) que indica el número (1 i(k) m) de la inecuación que, especializada en x_k, más discrepa con ser una solución. Empezamos la k+1-ésima iteración calculando la discrepancia general:

$$\theta_{k+1} = \min \left\{ \theta_k, \theta_k(x_k) \right\}$$

A continuación, lo que se pretende es trasladar el centro del elipsoide en la dirección i(k) y achicar el volumen del mismo tanto como se pueda sin dejar de encerrar al conjunto de soluciones.

Como paso previo, definimos:

$$F_k = -Q_k^t \times a_{i(k)}$$

$O(F_k)$: matriz ortogonal en $\mathbb{R}^{nxn}$ construida con el método de Gram-Schmidt con $\frac{F_k}{\|F_k\|}$ como primera columna

Λ_n: matriz diagonal en $\mathbb{R}^{nxm}$ cuyo vector diagonal es:

$$\left(\frac{n}{n+1}, \frac{n}{n^2-1}, \ldots\ldots\ldots, \frac{n}{n^2 - 1} \right)$$

El nuevo intento de solución y la elipse correspondiente se construyen ahora de la siguiente forma:

$$x_{k+1} = x_k + \frac{1}{n+1} \cdot \frac{Q_k \cdot F_k}{\|F_k\|}$$

$$Q_{k+1}=2^{1/8}n^2 . Q_k . O(F_k) . \wedge n$$

Se puede observar que el valor de la solución estimada X_k varía en una forma totalmente azarosa. Es, en realidad, el volumen del elipsoide (igual a det Q_k) lo que, al mantenerse suficientemente comprimido alrededor del conjunto de soluciones, asegura que se hallará una solución con suficiente velocidad.

El algoritmo finaliza si se produce una de las siguientes alternativas:

i) $\Theta_k(X_k) = 0$, en cuyo caso X_k es solución del sistema.

ii) el algoritmo ha ejecutado $N=16n^2L$ pasos sin que se verifique i). Si este es el caso se registra la discrepancia y el proceso termina con la conclusión de que el sistema es inconsistente.

3. SIMPLEX VERSUS KHACHIYAN

Dado un problema de Programación Lineal como el planteado por (1) en el inciso 2, el método SIMPLEX permite, como primera diferencia con el método de Khachiyan, que todos los coeficientes involucrados sean números reales sin ningún tipo de restricción, Khachiyan, en cambio, requiere que tanto los coeficientes de la matriz A como los del vector b, sean enteros. Esto no implica, en realidad, una pérdida de generalidad, ya que, en una computadora, cualquier número sólo se puede precisar hasta una cantidad fija de cifras decimales. De este modo, para los cálculos efectivos, siempre existe una potencia de diez apropiada tal que, multiplicando por ella a cada inecuación, podemos expresar todo el sistema con coeficientes enteros.

Otra <u>aparente</u> desventaja del algoritmo de K aplicado a resolver problemas de P.L. es que el algoritmo en si, cuya única función es la de resolver un sistema lineal de inecuaciones, requiere que todas las desigualdades de este sistema sean estrictas. Esto es necesario para que el conjunto de soluciones, si existe, sea abierto y por lo tanto tenga volumen positivo.

Sin embargo, se demuestra (ver [1] ó [2]) que el sistema $a_i x \leqslant b_i (i=1, m)$ tiene solución si y sólo si tiene solución el sistema $a_i x < b_i + 2^{-L}$ (i=1,m). Esta modificación del sistema de inecuaciones está considerada en el programa adjunto.

Continuando con el análisis del SIMPLEX, Dantzig demostró que si existe una solución óptima para (1), esta solución debe ser <u>básica</u>. Una solución básica es, geométricamente, un vértice del poliedro convexo determinado por las soluciones al sistema de restricciones. Numéricamente, una solución básica es aquella que tiene, a lo sumo, m coeficientes no nulos, donde m es la cantidad de inecuaciones del sistema. De esto se deduce que el número de soluciones básicas no puede superar $c = \binom{n}{m}$ que es la cantidad de formas de elegir m elementos de entre n.

Dantzig muestra cómo empezar desde una solución básica y luego moverse hacia otra mejor de tal forma que ninguna se repita. Por lo tanto, después de un número finito de iteraciones no superior a c, el proceso termina. Cuando lo hace, hemos encontrado una solución básica óptima. También puede detenerse el proceso cuando demuestra que el problema no es factible o la solución es ilimitada.

Se han realizado innumerables esfuerzos para demostrar que el método de Dantzig finaliza siempre después de una cantidad menor que c de iteraciones, pero no han dado resultado; por el contrario, se han generado problemas que requieren esta enorme cantidad de pasos para encontrar una solución.

Un número combinatorio se comporta esencialmente como 2^{nk} o e^{nk}, donde k es una constante. Por lo tanto, a medida que el número de variables, n, crece, la cantidad de iteraciones requeridas por el SIMPLEX será mucho mayor que la requerida por Khachiyan, que como vimos, crece polinomialmente con n (la cota superior era, en este caso, $N = 16\ n^2L$).

De esta manera, observamos que el SIMPLEX requiere, en principio, una cantidad de iteraciones que crece exponencialmente con el número de variables. Por otra parte, aunque la resolución del problema requiera menor cantidad de iteraciones, este número nunca se puede estimar debido al gran tamaño de la única cota superior conocida.

En cuanto a los cálculos efectivos, aunque el nivel de complejidad es parejo para ambos métodos, la gran precisión requerida por el agoritmo de Khachiyan hace que, en la práctica, el SIMPLEX siga siendo el método elegido hasta el momento.

4. PROGRAMA

En cuanto a la implementación del algoritmo, dejamos de lado, por el momento, el impedimento mayor que es la precisión requerida para hacerla.

Para transformar un problema de Programación Lineal en un problema de resolución de un sistema de inecuaciones lineales, hemos tenido que introducir la ecuación $c^t x - b^t y = 0$, descompuesta en las dos inecuaciones $c^t x - b^t y \quad 0$ y $-c^t x + b^t y \quad 0$.

Desde el punto de vista matemático, estamos considerando la intersección de dos semiespacios, es decir, exactamento el hiperplano determinado por la ecuación original. Pero, para los cálculos computacionales, debido a que los valores que se obtienen son precisos hasta cierta cantidad de cifras y nada más, es imposible calcular los puntos de la intersección sin caer en cierto margen de error que podría resultar significativo a medida que avanzan los cálculos.

Este problema se supera dando una tolerancia   0 y aceptando que $c^t x$ y $b^t y$ serán iguales cuando difieran en menos de . De este modo, reemplazando las dos inecuaciones mencionadas por $c^t x - b^t y$ y $-c^t x + b^t y$, lo que hemos hecho es rodear a $c^t x - b^t y = 0$ con un "tubo" de amplitud   en la dirección perpendicular a x.

Resumiendo, para un problema de Programación Lineal, el sistema completo de inecuaciones que resulta es:

$$\begin{pmatrix} A & 0_{mxm} \\ -I_n & 0_{nxm} \\ 0_{nxn} & -A^t \\ 0mxn & -I_m \\ c^t & -b^t \\ -c^t & b^t \end{pmatrix} . z \leqslant \begin{pmatrix} b \\ 0nx1 \\ -c \\ 0mx1 \\ \varepsilon \\ \varepsilon \end{pmatrix}$$

donde hemos notado con z al vector (x,y) de $^{n+m}$

Ahora debemos recurrir al algoritmo para decidir sobre la consistencia de un sistema de inecuaciones lineales y encontrar una solución, si existe. Nos encontramos entonces, en el paso inicial, con la necesidad de determinar una región que abarque todas las soluciones de dicho sistema.

Las soluciones, si existen, determinan un poliedro convexo cuyos vértices son las intersecciones de los hiperplanos que resultan de transformar en ecuaciones a las inecuaciones del sistema. El problema consiste, entonces, en estimar la distancia del origen al vértice más alejado.

Para no complicar la notación consideramos a nuestro sistema como $Ax \leqslant b$ con $A \in \mathbb{R}^{mxn}$ y $b \in \mathbb{R}^m$, ambos con coeficientes enteros. Sin pérdidas de generalidad, podemos suponer $m \geqslant n$ pues, de lo contrario, se pueden agregar inecuaciones triviales sin alterar la solución.

En estas condiciones, podemos aplicar la regla de Cramer para determinar todos los vértices. Para cada subconjunto de n ecuaciones del sistema Ax=b, los vértices serán

$$X_i = \frac{D_i}{D} \quad (i=1,n)$$

donde D es el determinante de la matriz de coeficientes y D_i es el determinante de la matriz obtenida con los mismos coeficientes pero reemplazando la i-ésima columna por b. Como trabajamos con coeficientes enteros, si $D \neq 0$, debe ser $|x_i| \leqslant |D_i|$ siendo $|x_i|$ la distancia del origen al vértice x_i. Por la desigualdad de Hadamard, $|D_i|$ es menor o igual que el producto de las normas de las columnas de la matriz en cuestión.

Esto explica por qué Khachiyan tomó 2^L como radio de la esfera inicial, ya que este valor es mayor que el producto de los valores absolutos de todos los coeficientes del sistema. Observamos, por otra parte, que no hace falta considerar un valor tan grande. En el programa cuyo listado está incluido en el presente trabajo, está contemplada la alternativa de elegir L de tal forma que cumpla con la condición requerida de que la esfera inicial contenga al conjunto de soluciones, pero que lo haga sin otro margen que el impuesto por la desigualdad de Hadamard. No obstante haber obtenido de esta forma un valor considerablemente menor para L, la precisión requerida (2^{-37nL}) ha variado sólo en forma despreciable.

4.1 Ventajas y desventajas del método de Khachiyan

A partir de lo expuesto, podemos plantear, como conclusión, que el método de Khachiyan cuenta con:

- Una asombrosa ventaja a nivel teórico por cuanto se puede determinar, desde el momento en que se conoce el problema, un límite superior del número de pasos necesario para alcanzar una solución (o, de lo contrario, decidir que tal solución no existe). A esto se suma, además, que este límite superior crece polinomialmente (y no exponencialmente) con el número de variables del problema. Esto nos permite vislumbrar la posibilidad de resolver en un tiempo razonable, problemas de dimensiones hasta ahora inconsiderables por el tiempo que demandaría su resolución.

- Una enorme desventaja en el orden práctico debida a que la precisión requerida en los cálculos para alcanzar dicho límite es de una magnitud tal, que casi la totalidad de las computadoras usadas hasta el momento no logran alcanzarla. Si con el tiempo se logra disminuir esta exigencia del algoritmo, no cabe duda que el método de Khachiyan reemplazará definitivamente al clásico método de Dantzig.

REFERENCIAS

1. P. GACS and L. LOVASZ, "Khachiyan s Algorithm for Linear Programming", Technical Report STAN-CS-79-750, Computer Science Department, Stanford University, 1979.

2. B. ASPVALL and R. F. STONE, (1979), Khachiyan's linear programming algorithm. Report CS 776, Computer Science Department, Stanford University.