Universidad Simón Bolívar
Departamento de Computación y T.I.
CI–5437 Inteligencia Artificial I
Abril–Julio 2010

Proyecto I – Búsqueda – 10 %

versión 1.2.1

Entrega: Jueves 8va, antes de las 8am.

Corre el año 2050. Nadie hubiera imaginado que pequeños signos como la elección de Obama y de Mockus[1] *iba a disparar la tecnificación en la toma decisiones de las democracia. En Venezuela, el Consejo Nacional Electoral maneja un importante fondo de investigación, y se ha asignado a los estudiantes del curso Inteligencia Artificial I implementar una regla de votación que permite computar resultados cuando los electores dan un orden de preferencia sobre un conjunto de opciones. Dicha regla fue introducida por Charles Lutwidge Dodgson, quien vivió en el siglo XIX.*[2] *El problema es que la regla de Dodgson es intratable. ¿Podrá usarse en la práctica a pesar de esa limitación?.*

Denotamos $a, b, c, d, \ldots$ como los nombres de un conjunto de opciones o candidatos[3]. La opinión de un votante lo representaremos por medio de una tupla que representa un orden total de preferencia entre los candidatos. Por ejemplo, $\langle b, c, d, a\rangle$ representa que b es preferido sobre c, c sobre d y d sobre a. Las opiniones de un conjunto de votantes se llaman un perfil, y se pueden representar resumidamente así:

3	5	3
a	d	d
c	c	c
d	a	b
b	b	a

Que significa que 3 votantes tienen una preferencia como la primera columna, donde a es preferido sobre c, etc. 5 votantes tienen preferencias como la segunda columna, y 3 como la tercera, para un total de 11 votantes.

Hay varias maneras de definir reglas de votación. Primero considere $N(x, y)$ como el número de votantes que prefieren al candidato x sobre el candidato y. En el ejemplo de arriba,

N(a,b) = 8, N(a,c) = 3, N(a,d) = 3,
N(b,a) = 3, N(b,c) = 0, N(b,d) = 0,
N(c,a) = 8, N(c,b) = 11, N(c,d) = 3,
N(d,a) = 8, N(d,b) = 11, N(d,c) = 8.

La regla de Condorcet dice que el ganador es un x tal que $N(x, y) > n/2$ para todo $y \neq x$, donde n es el número de votantes. En el ejemplo de arriba y vemos que d es el *Condorcet winner*.

En el siguiente ejemplo no hay Condorcet winner, pero d es es un Dodgson winner, que será definido más adelante.

[1] Nótese que el texto no aclara cuando eligieron a Mockus.
[2] Quizás le resulte familiar el pseudónimo usado por Dodgson al publicar sus libros de literatura: Lewis Carroll.
[3] En las pruebas se usará un máximo de 250 candidatos.

10	8	7	4
c	d	d	b
b	a	b	a
a	b	a	c
d	c	c	d

Llamemos un *cambio elemental* al intercambio de dos posiciones contiguas en una preferencia de un votante. Ej: de la preferencia $\langle a, b, c, d\rangle$ a $\langle a, d, b, c\rangle$ hay dos cambios elementales. Sea $D(x)$ como el menor número de cambios elementales, tal que x es el Condorcet winner. La regla de *Dodgson* establece como ganador al x con menor $D(x)$.

Actividades:

1. Encontrar el Dodgson winner mediante búsqueda en amplitud (BFS).

   - Un estado es un perfil.
   - De cada estado, los sucesores corresponden a intercambiar en alguna preferencia de algún votante para obtener un nuevo perfil. Note que al modificar la preferencia de un sólo votante, esa nueva preferencia puede ser distinta de todas las otras en el perfil.
   - Los nodos metas son los que tienen un Condorcet winner.
   - Implemente búsqueda en amplitud para encontrar un Dodgson winner. Búsqueda en amplitud permite encontrar un camino de costo mínimo a una meta. Por lo tanto, una meta de costo mínimo corresponder exactamente a un Dodgson winner.

2. Dado el gran número de estados sucesores al considerar cambios elementales de un perfil, un algoritmo de búsqueda en amplitud agotará rápidamente la memoria. Se quiere usar un algoritmo IDA*, que permitirá usar memoria lineal.

   - Déficit es el número de agentes adicionales que deben preferir x sobre y, de manera que x triunfe sobre y en una elección entre los dos. Más formalmente:

     $$defct(x, y) = max(0, \left\lceil \frac{n+1}{2} \right\rceil - N(x, y)).$$

     Observe que si $N(x, y)$ ya es suficiente para ganar, entonces $defct(x, y)$ retorna 0.
   - Sea $T(x) = \sum_{y \neq x} defct(x, y)$. Observe que $T(x)$ suma los cambios necesarios para que x triunfe sobre cada $y \neq x$, convirtiéndolo en un Condorcet winner. Pero $defct(x, y)$ no cuenta el número de cambios elementales reales, por lo que $T(x)$ es una cota inferior.
   - Pero esa cota es mejorable. Puede probarse [a] que $T'(x) \leq D(x)$, donde

     $$T'(x) = \frac{T(x) + \log m + 1}{\log m + 3}$$

     donde m es el número de opciones para votar y log es logaritmo base dos.
   - Use $T'(x)$ para construir una función heurística en un algoritmo IDA* para encontrar el Dodgson winner. Compare su rendimiento con el algoritmo anterior.

Requerimientos técnicos:

1. Puede implementarlo en el lenguaje que quiera, siempre que sea un lenguaje compilado.

2. Su proyecto debe funcionar en las maquinas Linux del LDC, incluyendo la versión del lenguaje de programación que esté allí instalada.

3. Su ejecutable debe llamarse `dodgson`. Este recibirá como parámetro el algoritmo a usar, indicando `-ida` o `-bfs`. Opcionalmente recibirá el parámetro `-all`, para indicar que se quiere obtener todos los Dodgson winners. Opcionalmente recibirá el parámetro `-final <archivo>`, para indicar que en el archivo `<archivo>` debe escribirse el perfil final. En caso de que se esté usando la opción `-all`, entonces para cada Dodgson winner `A`, su perfil final se guardará en el archivo `A-<archivo>`. El último parámetro será siempre el archivo con el perfil inicial.

   - Una llamada a su programa

     ```
     ./dodgson -ida dodgson1.txt
     ```

     debe retornar una salida así:

     ```
     Dodgson winner: d
     Num cambios elementales: 4
     Nodos generados: n
     Nodos expandidos: m
     ```

     donde `n` son todos los nodos generados por el algoritmo, y `m` son los nodos cuyos hijos fueron generados. Esto indica que luego de 4 cambios elementales sobre el perfil original, d es un Condorcet winner.

   - Si llama a su programa con `-all`

     ```
     ./dodgson -ida -all dodgson1.txt
     ```

     debe retornar una salida así:

     ```
     Dodgson winners: d e
     Num cambios elementales: 4
     Nodos generados: n
     Nodos expandidos: m
     ```

     Que indica que ambos `d` y `e` son Dodgson winners porque cada uno, tras 4 cambios elementales, pueden convertirse en un Condorcet winner. Evidentemente dichos cambios elementales serán diferentes para `d` y `e`. Desde el punto de vista de la implementación es una pequeña variación sobre los algoritmos BFS y IDA*.

4. El formato de archivo para leer o escribir un perfil es el siguiente:

   ```
   profile
   4 a b c d
   4
   10 c b a d
   8 d a b c
   7 d b a c
   4 b a c d
   ```

Este archivo codifica el segundo perfil de este enunciado. La 2da linea indica que hay cuatro opciones de voto, y a continuación da los nombres de dichas opciones en orden no especificado. Cada opción es una cadena de caracteres alfabéticos. La 3ra línea indica cuantas líneas con preferencias siguen en el archivo. Luego cada linea posterior corresponde a una columna de un perfil indicando, número de votantes con la preferencia que sigue. La opción más a la izquierda es la más preferida.

5. Debe entregar un archivo `.tar.gz` que contenga al menos:

   informe.pdf que describa muy resumidamente las decisiones tomadas para hacer su proyecto. No explique los algoritmos IDA* ni BFS, ni cosas que estén explicadas en este enunciado. Debe comentar sus decidiones sobre estructuras de datos, los resultados de su comparación y una pequeña discusión sobre sus resultados, y recomendaciones sobre como mejorarlo.

   README.txt que indique los requerimientos técnicos de su programa y otra información relevante sobre los archivos del `.tar.gz`

   Makefile tal que `make` compile su programa.

Alguno comentarios sobre la implementación:

1. Cuando arriba se define un Estado y como se expande, es sólo para definir como debe ser el espacio de búsqueda. Su implementación puede hacer cualquier cosa que genere el mismo espacio de búsqueda.

2. Por ejemplo, ir modificando desde el perfil inicial puede llegar a ocupar mucho espacio. Otra posibilidad es guardar sólo la información necesaria en cada nodo para responder preguntas sobre el estado, sin tener que tener el estado explícitamente, sino refiriéndose a los nodos padres. Piense en que necesita hacer con cada nodo para expandirlo, y para evaluar su heurística.

Referencias:

a. I. Caragiannis, C. Kaklamanis, N. Karanikolas and A. Procaccia. *Socially Desirable Approximations for Dodgson's Voting Rule*. 11th ACM Conference on Electronic Commerce. Junio 2010, Massachusetts, USA.