Decisões em Domínios Combinatórios: Preferências decomponíveis

Sergio Queiroz

srmq@cin.ufpe.br

20 de março de 2009

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Na aula passada...

- Os objetos que representavam as alternativas de um problema de decisão eram elementos “indivisíveis”
  - Candidatos
  - Projetos
- No entanto, temos situações onde os objetos podem ser descritos por uma coleção de atributos
  - Exemplos:
    - Automóvel
      - Marca, cor, número de portas, motorização, acessórios diversos etc.
    - Viagem
      - Destino, tipo de acomodação, meio de transporte, atividades
    - Almoço
      - Entrada, prato principal, sobremesa, bebida

Decisão: domínios combinatórios

- Espaço de alternativas = produto cartesiano
  - $X = X_1 \times X_2 \times ... \times X_n$ X_i domínio finito de valores possíveis
  - Se cada domínio de atributo tem p valores, tamanho p^n
  - Representar preferências em extensão necessita de um grande espaço em memória
    - n = 10, p = 10 → 10 GB; n = 10, p = 20 → 10 TB
- A tomada de decisão é muito difícil (alternativas demais)

Solução:

- Explorar a decomponibilidade das preferências (independências)
- Representações compactas

Benefícios:

- Extração das preferências mais simples
- Utilização eficaz do espaço de memória
- Algoritmos eficazes de exploração

Funções de utilidade decomponíveis

- Espaço de alternativas: $X = X_1 \times X_2 \times ... \times X_n$
- Apreciação (utilidade) de uma alternativa $x \in X$
  - $u : X \mapsto \mathbb{R}$
- Modelo aditivo $u(x) = \sum_{i=1}^{n} u_i(x_i)$
  - Simples e eficaz: $p^n \rightarrow p \times n$
  - Independência entre atributos

Exemplo: compra de bilhetes de trem

- Bilhetes de trem: $A = \{h, c, v\}$
  - h (horário da partida): $h \in$ {noturno (h_1), diurno (h_2)}
  - c (categoria): $c \in$ {cabine (c_1), assento (c_2)}
  - v (velocidade): $v \in$ {trem-bala (v_1), trem normal (v_2)}
- Preferências de um usuário
  - lexicográficas
    - Maior prioridade, viajar à noite: $h_1 >_1 h_2$
    - Em seguida, viajar numa cabine: $c_1 >_2 c_2$
    - Finalmente, viajar no trem-bala: $v_1 >_3 v_2$
  - O que implica as preferências
    - $(h_1, c_1, v_1) > (h_1, c_1, v_2) > (h_1, c_2, v_1) > (h_1, c_2, v_2) > (h_2, c_1, v_1) > (h_2, c_1, v_2) > (h_2, c_2, v_1) > (h_2, c_2, v_2)$

Exemplo: compra de bilhetes de trem (2)

- Preferências de um usuário
  - lexicográficas
    - Maior prioridade, viajar à noite: $h_1 >_1 h_2$
    - Em seguida, viajar numa cabine: $c_1 >_2 c_2$
    - Finalmente, viajar no trem-bala: $v_1 >_3 v_2$
  - O que implica as preferências
    - $(h_1, c_1, v_1) > (h_1, c_1, v_2) > (h_1, c_2, v_1) > (h_1, c_2, v_2) > (h_2, c_1, v_1)$ $> (h_2, c_1, v_2) > (h_2, c_2, v_1) > (h_2, c_2, v_2)$
  - Podem ser modelizadas por uma função de utilidade aditiva
    - $u(h, c, v) = u_1(h) + u_2(c) + u_3(v)$
    - Exemplo: $u_1(h_1) = 100$, $u_1(h_2) = 90$; $u_2(c_1) = 10$, $u_2(c_2) = 5$; $u_3(v_1) = 1$, $u_3(v_2) = 0$.
      - (h_1, c_1, v_1) (111) $> (h_1, c_1, v_2)$ (110) $> (h_1, c_2, v_1)$ (106) $> (h_1, c_2, v_2)$ (105) $> (h_2, c_1, v_1)$ (101) $> (h_2, c_1, v_2)$ (101) $> (h_2, c_2, v_1)$ (96) $> (h_2, c_2, v_2)$ (95)

Definição: Independência preferencial

- Espaço de alternativas: $A = A_1 \times A_2 \times ... \times A_n$
- Seja $W = \{w_1, ..., w_k\}$ um subconjunto de $\{1, ..., n\}$
- Um subconjunto de atributos A_W é preferencialmente independente de seu complemento A_{-W} (escrevemos $PI(A_W, A_{-W})$) ssi, para todo $a_W, a'_W \in A_W$; $a_{-W}, a'_{-W} \in A_{-W}$:

$$(a_W, a_{-W}) \geq (a'_W, a_{-W}) \Leftrightarrow (a_W, a'_{-W}) \geq (a'_W, a'_{-W})$$

  - Preferências sobre atributos com índices em W permanecem as mesmas quando fixamos os valores dos outros atributos, pouco importa os valores nos quais fixamos os valores desses outros atributos.
    - a_W é preferível a a'_W *ceteris paribus* (mantidas inalteradas todas as outras coisas)

Função de utilidade aditiva ⇒ cada atributo é independente

Independência preferencial - Atenção!

- Independência preferencial não é necessariamente bidirecional
  - Podemos ter um subconjunto de atributos A_w que é preferencialmente independente de seu complemento A_{-w}, ao mesmo tempo em que A_{-w} não é preferencialmente independente de A_w
  - Exemplo (Avaliação de automóveis):
    - Atributos: Transmissão (T) e Marca (M)
      - T = {automática, manual}; M = {audi, citroën}
    - Preferências
      - (audi, automática) > (audi, manual) > (citroën, manual) > (citroën, automática)
    - *PI*(*M*, *T*)
    - Mas !*PI*(*T*, *M*)

Definição: Independência condicional

- Espaço de alternativas: $A = A_1 \times A_2 \times ... \times A_n$
- W, Y e Z três conjuntos não-nulos que particionam $\{1, ..., n\}$
  - i.e., $W \cup Y \cup Z = \{1, ..., n\}$; $W \cap Y = W \cap Z = Y \cap Z = \emptyset$
- A_W é condicionalmente independente de A_Y dado A_Z (escrevemos $CPI(A_W, A_Z, A_Y)$) ssi, para todo a_W, a'_W, a_Y, a'_Y, a_Z, temos que:

$$(a_W, a_Z, a_Y) \geq (a'_W, a_Z, a_Y) \Leftrightarrow (a_W, a_Z, a'_Y) \geq (a'_W, a_Z, a'_Y)$$

  - As preferências sobre os valores de atributos com índices W são preferencialmente independentes dos valores dos atributos com índices Y uma vez que fixamos os valores dos atributos com índice Z

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Modelo Aditivo Generalizado

Generalized Additive Independence (GAI)

- Definição (Fishburn, 70; Baccus e Groove 95)
  - $X = X_1 \times X_2 \times ... \times X_n$
  - $C_1, ..., C_k$ subconjuntos de $N = \{1, ..., n\}$ tais que

$$N = \bigcup_{i=1}^{k} C_i$$

  - $X_{C_i} = \{X_j : j \in C_i\}$; $u_i : X_{C_i} \rightarrow \mathbb{R}$
  - Uma função de utilidade GAI sobre X pode ser escrita na forma:

$$u(x_1, ..., x_n) = \sum_{i=1}^{k} u_i(x_{C_i})$$

Exemplo:

$u(x_1, x_2, x_3, x_4) = u_1(x_1) + u_2(x_2, x_3) + u_3(x_3, x_4)$

Dependências entre atributos

Fatores não-disjuntos

Bilhetes de trem, novas preferências

- Bilhetes de trem: $A = \{h, c, v\}$
  - h (horário da partida): $h \in$ {noturno (h_1), diurno (h_2)}
  - c (categoria): $c \in$ {cabine (c_1), assento (c_2)}
  - v (velocidade): $v \in$ {trem-bala (v_1), trem normal (v_2)}
- O mais importante é ter bilhetes adaptados ao tipo de viagem.
  - Diurnas ⇒ assento; Noturnas ⇒ cabine: $h_2: c_2 > c_1$, $h_1: c_1 > c_2$
- Em seguida, prefiro viajar à noite: $h_1 > h_2$
- Finalmente, pela manhã prefiro o trem-bala, mas à noite prefiro o normal: $h_2: v_1 > v_2$, $h_1: v_2 > v_1$
- $(h_1, c_1, v_2) > (h_1, c_1, v_1) > (h_2, c_2, v_1) > (h_2, c_2, v_2) > (h_1, c_2, v_2) > (h_1, c_2, v_1) > (h_2, c_1, v_1) > (h_2, c_1, v_2)$

Bilhetes de trem, novas preferências (2)

- $(h_1, c_1, v_2) > (h_1, c_1, v_1) > (h_2, c_2, v_1) > (h_2, c_2, v_2) > (h_1, c_2, v_2) > (h_1, c_2, v_1) > (h_2, c_1, v_1) > (h_2, c_1, v_2)$
- Esta ordem pode ser modelizada por funções GAI-decomponíveis
  - $u(h, c, v) = u_1(h, c) + u_2(h, v)$
  - Exemplos de valores de utilidade

$u_1(h, c)$	c_1	c_2
h_1	110	10
h_2	0	100

$u_1(h, v)$	v_1	v_2
h_1	0	5
h_2	5	0

115 110 105 100 15

- $(h_1, c_1, v_2) > (h_1, c_1, v_1) > (h_2, c_2, v_1) > (h_2, c_2, v_2) > (h_1, c_2, v_2) > (h_1, c_2, v_1) > (h_2, c_1, v_1) > (h_2, c_1, v_2)$

10 5 0

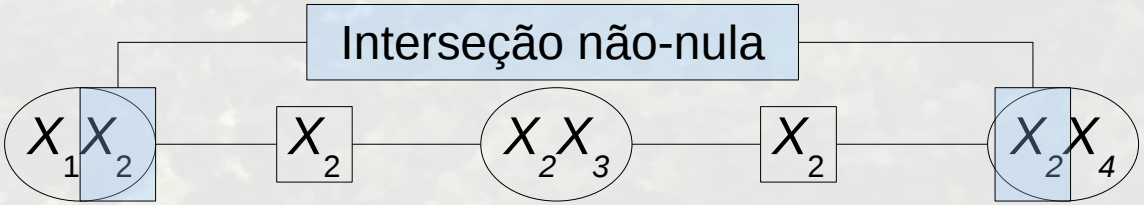
Subconjuntos não são PI - Atenção!

- Em geral, os subconjuntos de atributos nos termos de utilidade u_i não representam independências preferenciais entre os atributos de u_i e seu complemento.
- Exemplo
  - $u(a, b, c) = u_1(a, b) + u_2(b, c)$ (domínios binários)
    - $u_1(a_0, b_0) = u_1(a_1, b_1) = 10$
    - $u_1(a_0, b_1) = u_1(a_1, b_0) = 1$
    - $u_2(b_0, c_1) = u_2(b_1, c_0) = 100$
    - $u_2(b_1, c_1) = u_2(b_0, c_0) = 9$
  - $\{a, b\}$ não é preferencialmente independente de $\{c\}$
    - $(a_0, b_0, c_1) \geq (a_0, b_1, c_1)$ $\quad 110 > 10$
    - $(a_0, b_1, c_0) \geq (a_0, b_0, c_0)$ $\quad 101 > 19$

Portanto não é possível construir/otimizar os termos separadamente

Redes GAI

- $u(x_1, x_2, x_3, x_4) = u_1(x_1, x_2) + u_2(x_2, x_3) + u_3(x_2, x_4)$

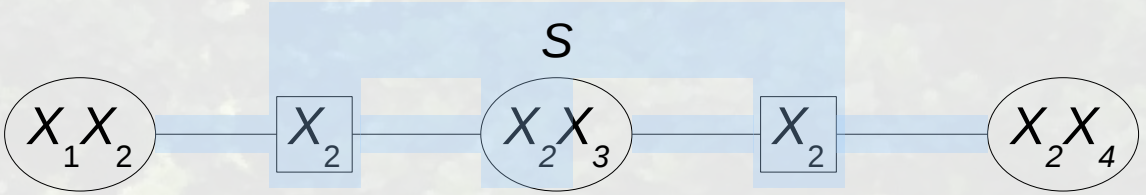


elipse = clique retângulo = separador

- Árvores de junção como aquelas utilizadas na literatura de redes bayesianas
- Propriedade de interseção corrente (*running intersection*)
  - Para todo par de cliques (C_1, C_2) com uma interseção não-nula S

Redes GAI

- $u(x_1, x_2, x_3, x_4) = u_1(x_1, x_2) + u_2(x_2, x_3) + u_3(x_2, x_4)$



elipse = clique retângulo = separador

- Árvores de junção como aquelas utilizadas na literatura de redes bayesianas
- Propriedade de interseção corrente (*running intersection*)
  - Para todo par de cliques (C_1, C_2) com uma interseção não-nula *S*, *S* é um subconjunto de toda clique e todo separador no caminho entre C_1 e C_2
    - uma variável pertencente a uma clique externa *C* e que não pertence à clique vizinha de *C* não aparece em nenhuma outra clique da rede GAI.

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

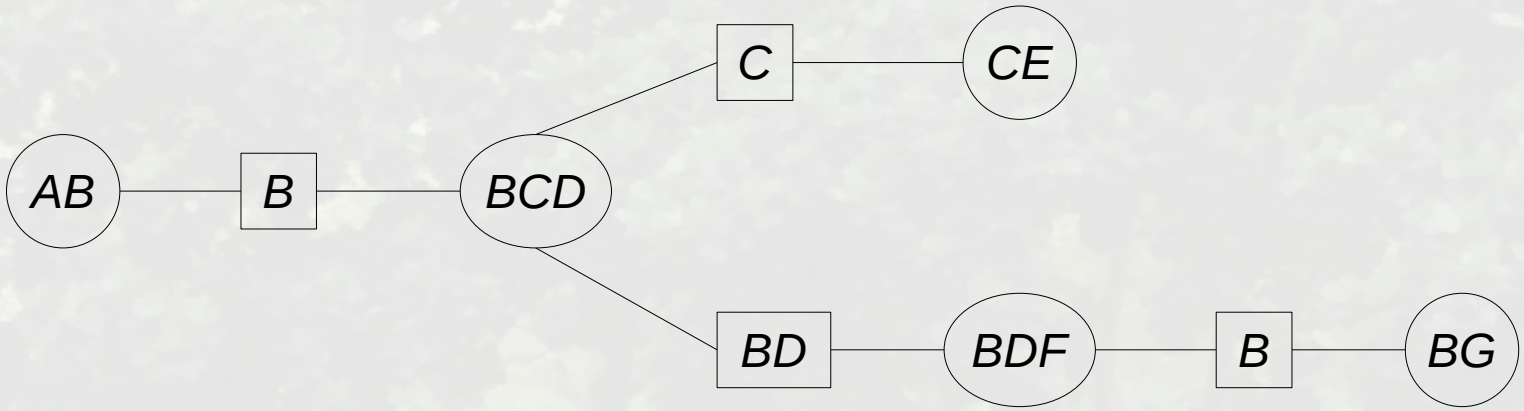
Otimização

- Maximização de $u(X_1, ..., X_n)$ no espaço $X = X_1 \times ... \times X_n$.
- Propriedades úteis para resolver de forma eficaz o problema:
  - o max em $X_1 \times X_2 \times ... \times X_n$ de $u(X_1, ..., X_n)$ pode ser decomposto como $\max_{X_1} \max_{X_2} ... \max_{X_n} u(X_1, ..., X_n)$ onde a ordem dos max não tem nenhuma importância;
  - se $u(X_1, ..., X_n)$ puder ser decomposta como $f() + g()$ onde $f()$ não depende de X_i então $\max_{X_i}[f() + g()] = f() + \max_{X_i} g()$;
  - em uma rede GAI, a interseção corrente garante que uma variável que pertence a uma clique externa *C* e que não pertence à clique vizinha de *C* não aparece em nenhuma outra clique da rede GAI.

Otimização: estratégia de maximização

- Para calcular a utilidade maximal
  - Processo em duas fases: coleta e difusão
    - Coleta
      - Maximização das cliques exteriores às interiores
      - Os resultados são transmitidos às cliques internas vizinhas e as cliques exteriores são eliminadas
    - Difusão
      - Fase de atribuição de valores (distribuição)
      - Propagação no sentido inverso da coleta
- Estratégia desenvolvida no contexto da programação dinâmica (Bertelé et Brioschi 72)
  - Reutilizada em vários domínios: **bancos de dados** (Beeri et al 83), **redes bayesianas** (Lauritzen and Spiegelhalter 88), **CSP** (Dechter et Pearl 89), **diagramas de influência** (Shenoy et Shafer 90), **soft constraints** (Bistarelli et Rossi 95)

Otimização



- Expressão da melhor configuração

$$\max_{A,\,B,\,C,\,D,\,E,\,F,\,G} u_1(A, B) + u_2(C, E) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

$$\max_A \max_B \max_C \max_D \max_E \max_F \max_G u_1(A, B) + u_2(C, E) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

Otimização: fase de coleta (1/5)

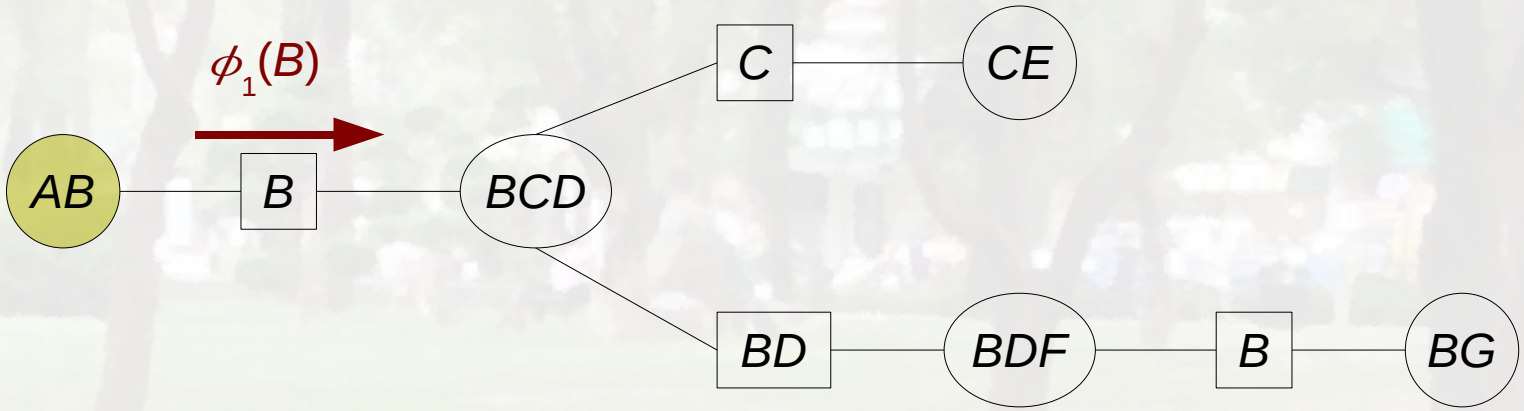
- Eliminação de *A*

$$\max_B \max_C \max_D \max_F \max_G \max_E \max_A$$

$$u_1(A, B) + u_2(C, E) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

$$= \max_B \max_C \max_D \max_F \max_G \max_E$$

$$\phi_1(B) + u_2(C, E) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$



Otimização: fase de coleta (2/5)

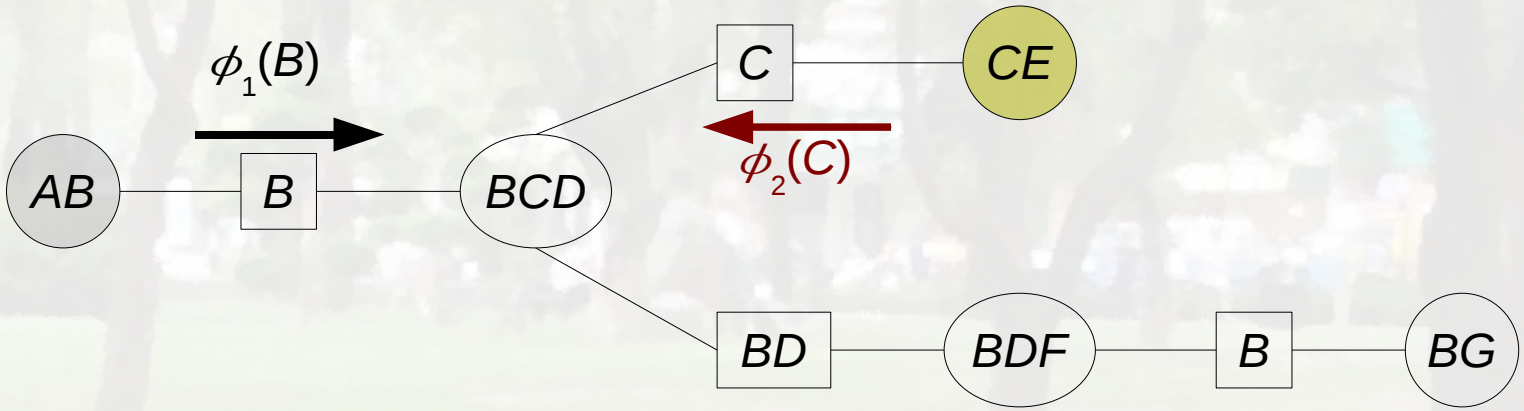
- Eliminação de *E*

$$\max_B \max_C \max_D \max_F \max_G \max_E$$

$$\phi_1(B) + u_2(C, E) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

$$= \max_B \max_C \max_D \max_F \max_G$$

$$\phi_1(B) + \phi_2(C) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

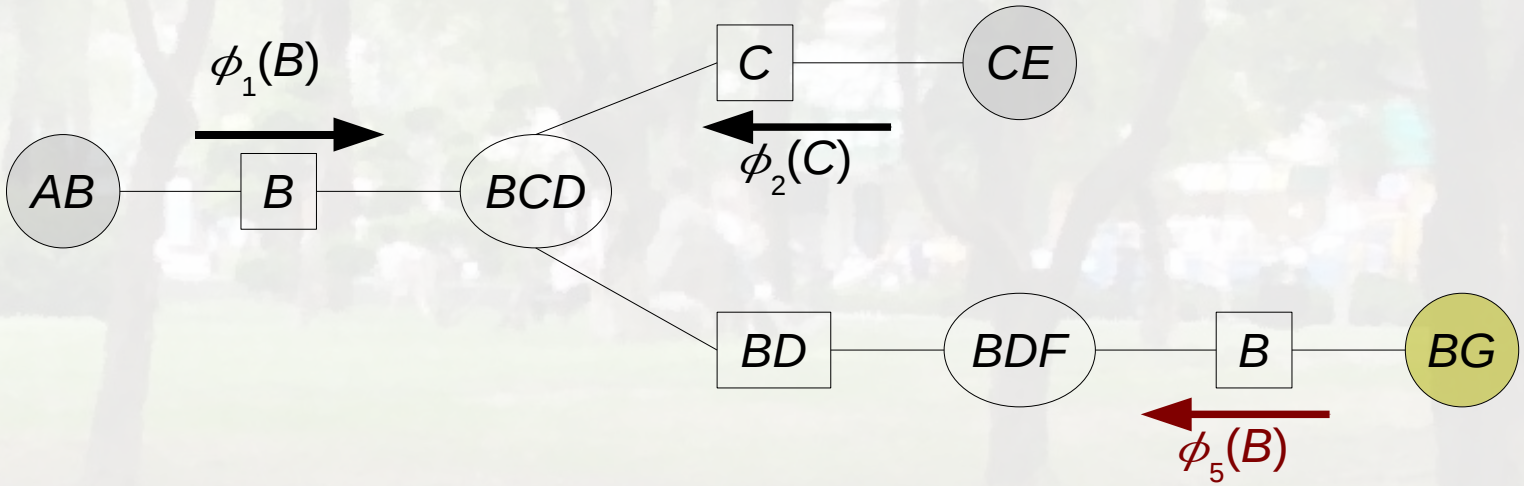


Otimização: fase de coleta (3/5)

- Eliminação de G

$$\max_B \max_C \max_D \max_F \max_G$$
$$\phi_1(B) + \phi_2(C) + u_3(B, C, D) + u_4(B, D, F) + u_5(B, G)$$

$$= \max_B \max_C \max_D \max_F$$
$$\phi_1(B) + \phi_2(C) + u_3(B, C, D) + u_4(B, D, F) + \phi_5(B)$$



Otimização: fase de coleta (4/5)

- Eliminação de *F*

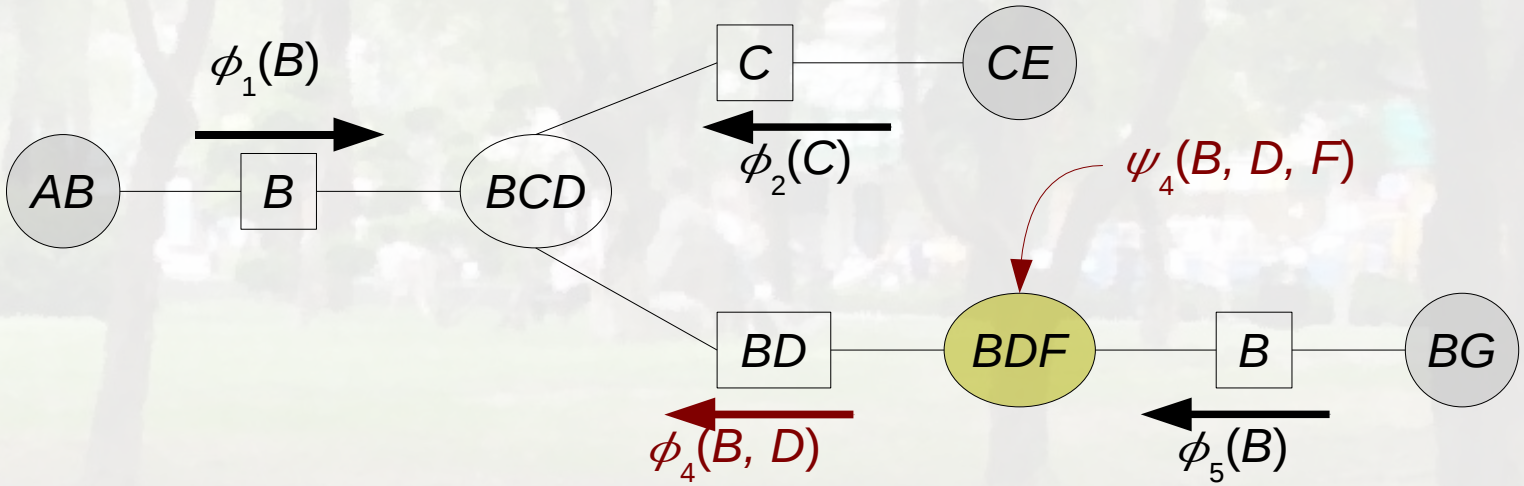
$\max_B \max_C \max_D \max_F$

$\phi_1(B) + \phi_2(C) + u_3(B, C, D) + \psi_4(B, D, F)$

où $\psi_4(B, D, F) = u_4(B, D, F) + \phi_5(B)$

$= \max_B \max_C \max_D$

$\phi_1(B) + \phi_2(C) + u_3(B, C, D) + \phi_4(B, D)$



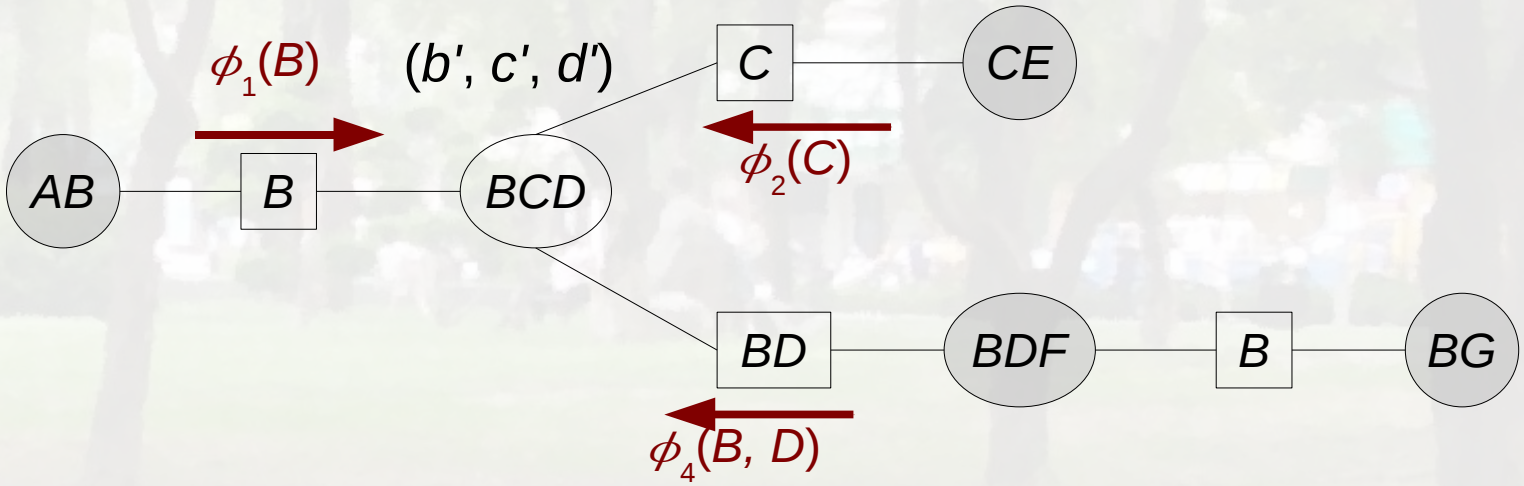
Otimização: fase de coleta (5/5)

- Eliminação de *B, C, D*

  $\max_B \max_C \max_D \psi_3(B, C, D)$

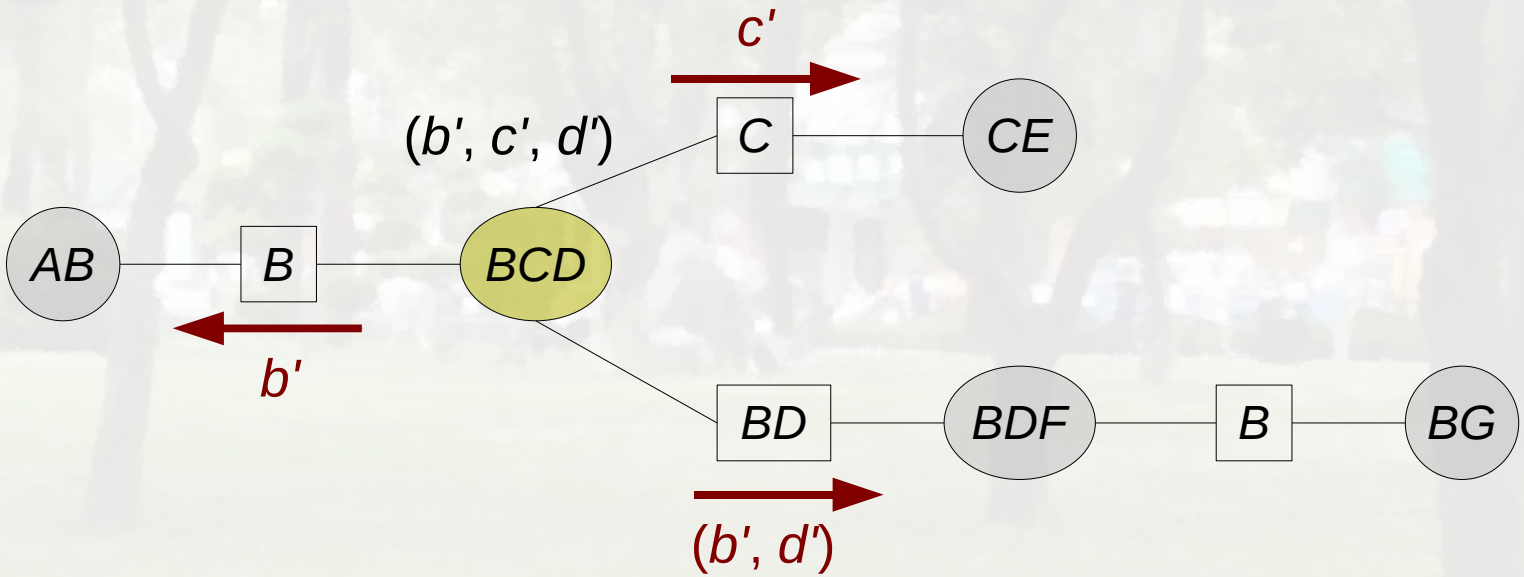
  où $\psi_3(B, C, D) = \phi_1(B) + \phi_2(C) + \phi_4(B, D) + u_3(B, C, D)$

- $\text{Argmax}_{B,C,D}\ \psi_3(B, C, D) = (b', c', d')$



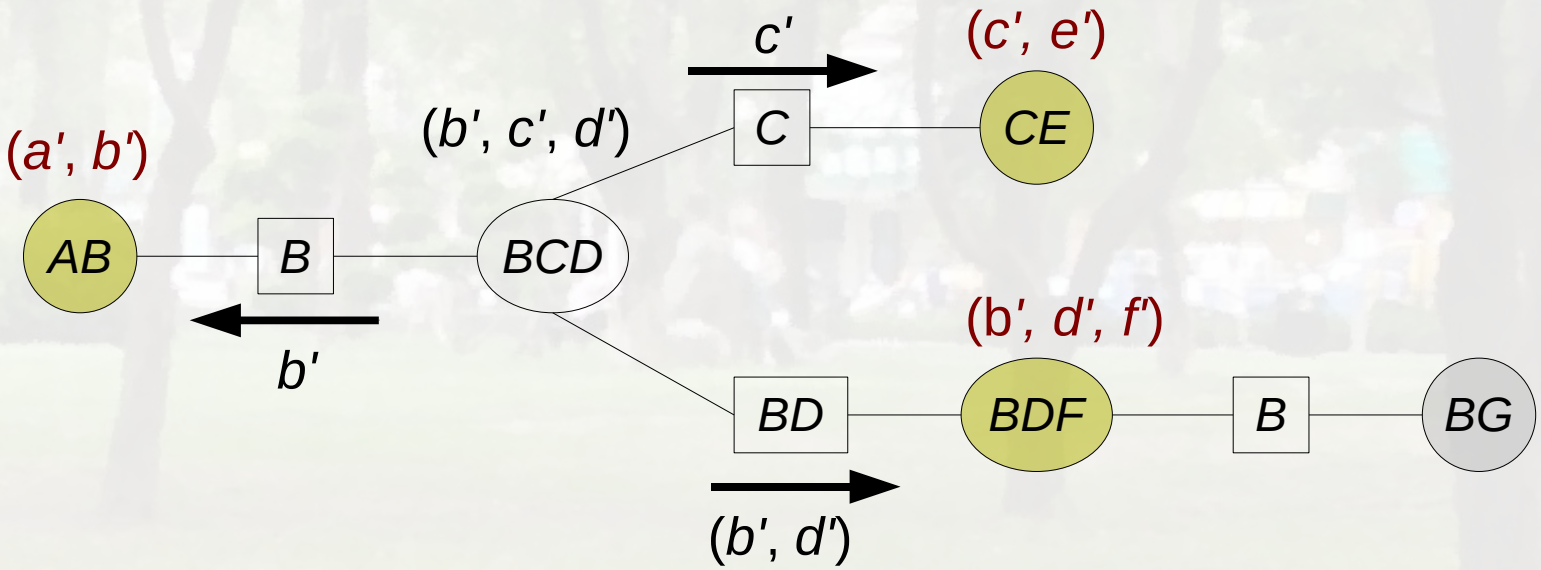
Otimização: fase de difusão (1/4)

- Distribuição pela clique *BCD*
  - *BCD* envia mensagens aos separadores vizinhos com os valores de sua tupla ótima



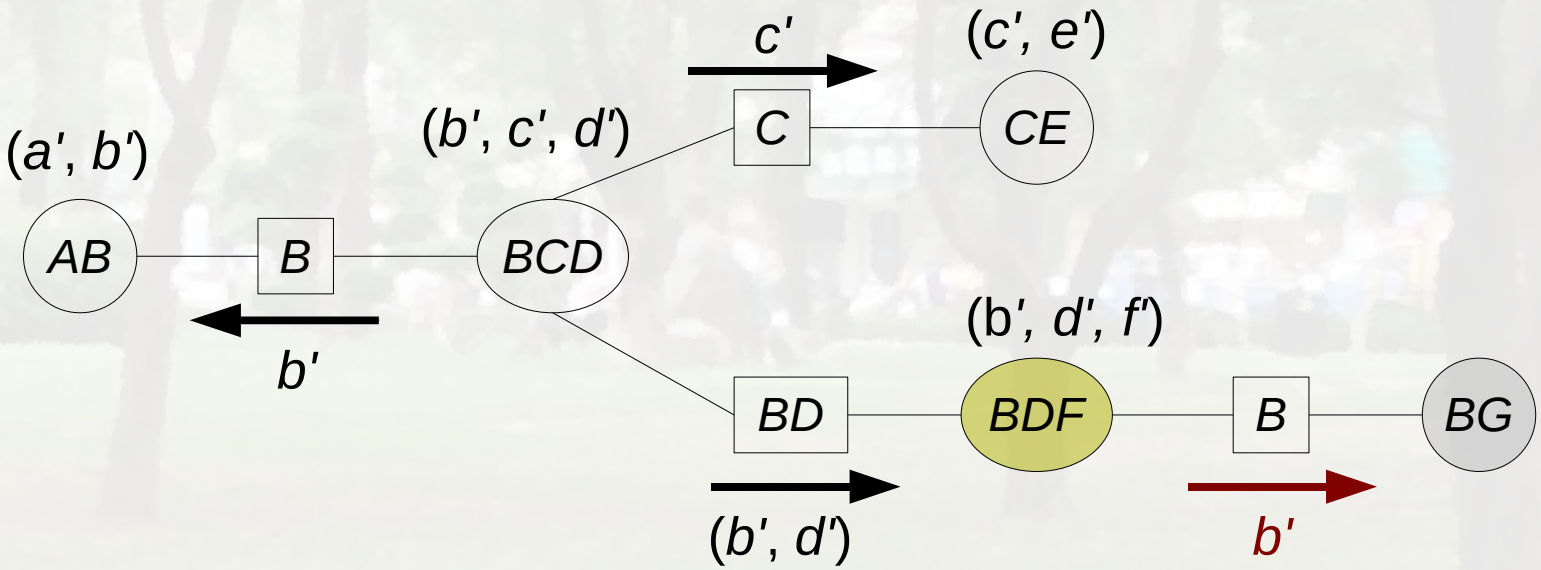
Otimização: fase de difusão (2/4)

- Propagação pelos separadores
  - Clique AB → $a' = \text{Argmax}_A \, u_1(A, b')$
  - Clique CE → $e' = \text{Argmax}_E \, u_2(c', E)$
  - Clique BDF → $f' = \text{Argmax}_F \, \psi_4(B, D, F)$



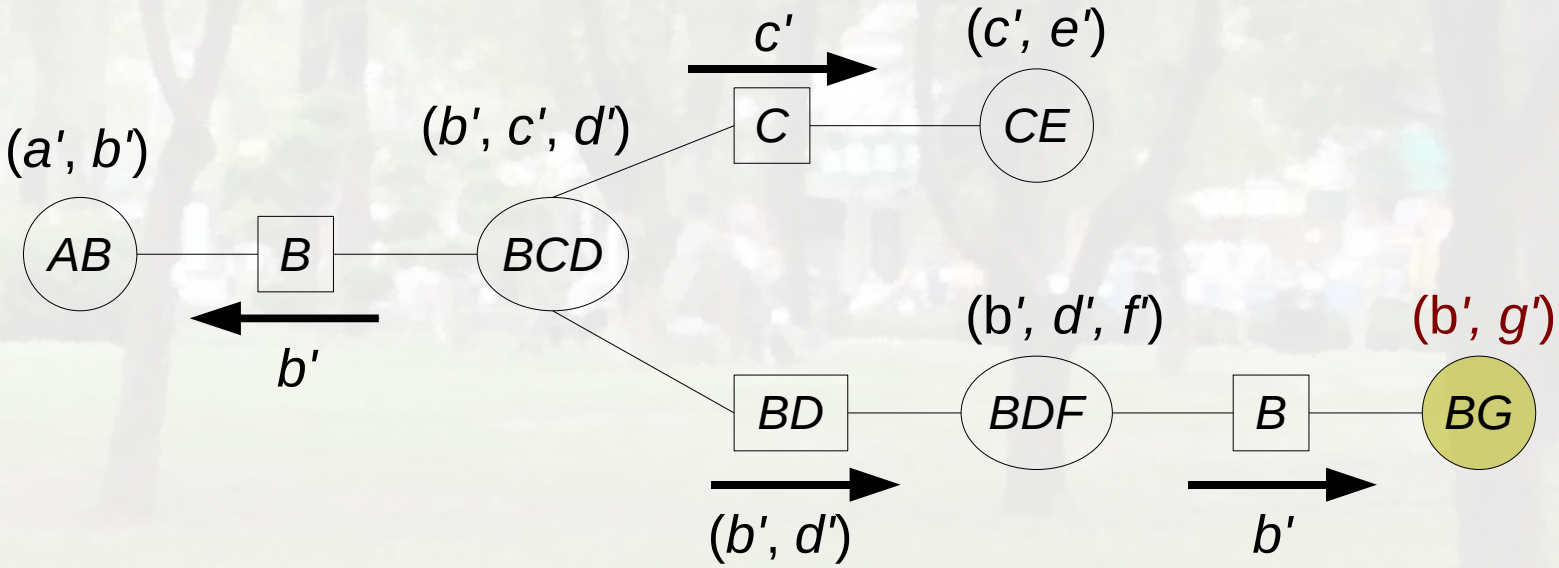
Otimização: fase de difusão (3/4)

- Distribuição pela clique *BDF*
  - *BDF* envia uma mensagem ao separador *B* com o valor ótimo *b'*

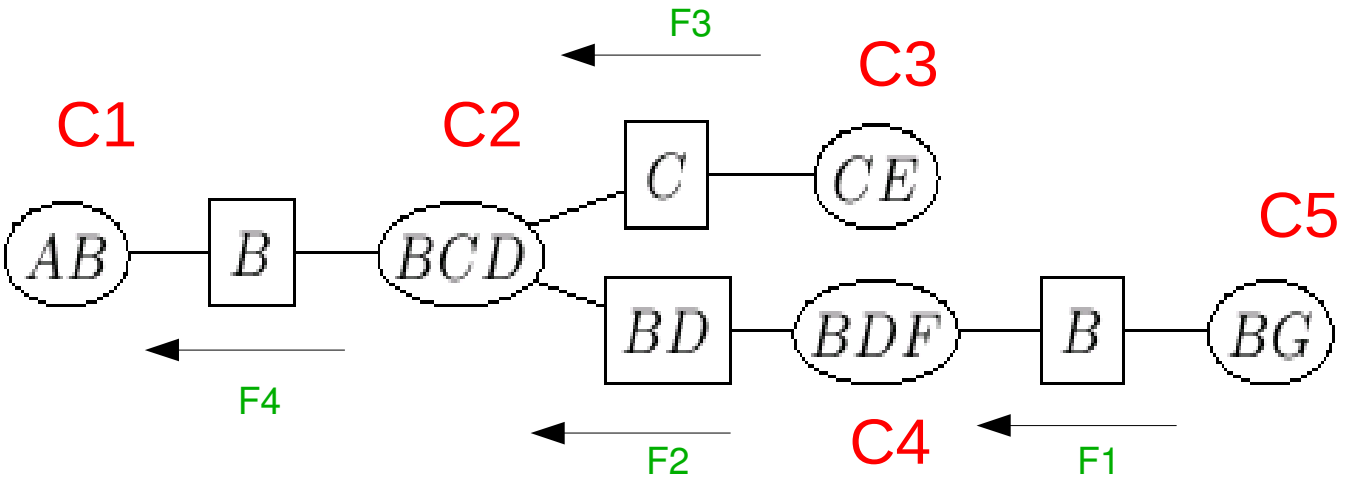


Otimização: fase de difusão (4/4)

- Propagação pelo separador *B*
  - Clique *BG* → g′ = $\text{Argmax}_B\ u_5(b', G)$
- Configuração ótima
  - (*a′*, *b′*, *c′*, *d′*, *e′*, *f′*, *g′*)



Uma outra ordem possível...



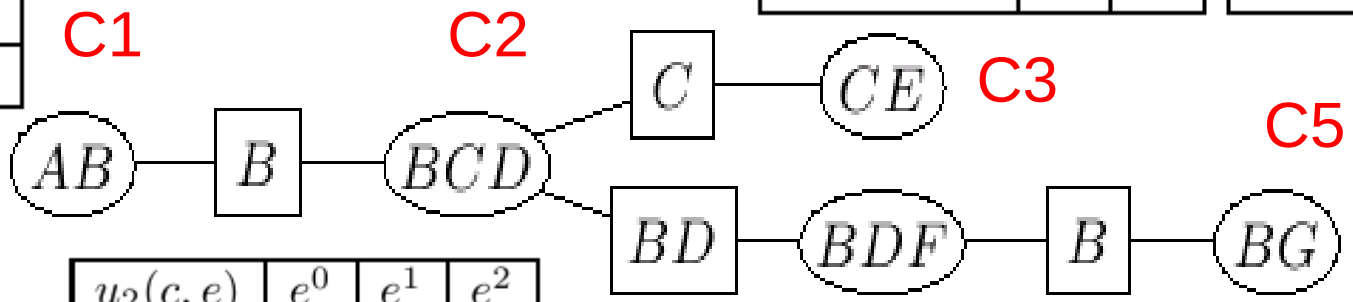
Fluxos na direção da raiz da árvore

Antes da coleta...

$u_1(a,b)$	b^0	b^1
a^0	8	2
a^1	4	3
a^2	1	7

$u_3(b^0,c,d)$	d^0	d^1
c^0	0	2
c^1	5	1

$u_3(b^1,c,d)$	d^0	d^1
c^0	7	1
c^1	2	4



$u_2(c,e)$	e^0	e^1	e^2
c^0	6	3	5
c^1	3	4	0

$u_5(b,g)$	g^0	g^1
b^0	0	9
b^1	6	4

C4

$u_4(b^0,d,f)$	f^0	f^1
d^0	4	2
d^1	3	8

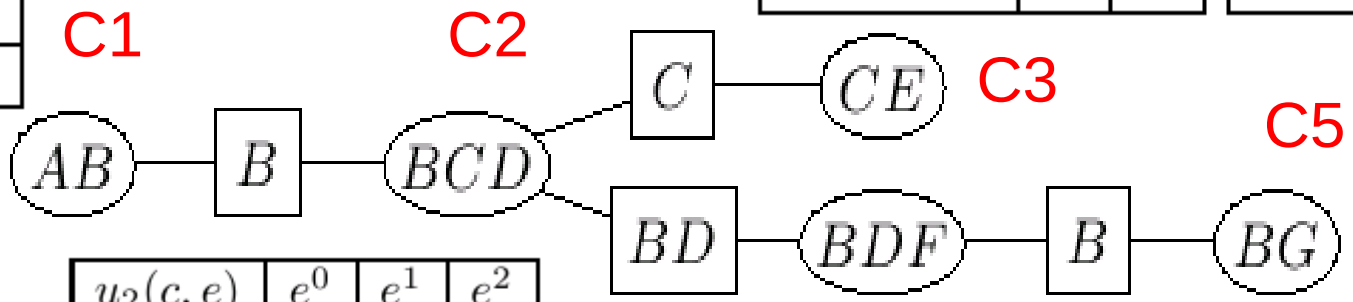
$u_4(b^1,d,f)$	f^0	f^1
d^0	5	8
d^1	9	0

Iniciando a coleta...

$u_1(a,b)$	b^0	b^1
a^0	8	2
a^1	4	3
a^2	1	7

$u_3(b^0,c,d)$	d^0	d^1
c^0	0	2
c^1	5	1

$u_3(b^1,c,d)$	d^0	d^1
c^0	7	1
c^1	2	4



$u_2(c,e)$	e^0	e^1	e^2
c^0	6	3	5
c^1	3	4	0

$u_5(b,g)$	g^0	g^1
b^0	0	9
b^1	6	4

C4

$u_4(b^0,d,f)$	f^0	f^1
d^0	4	2
d^1	3	8

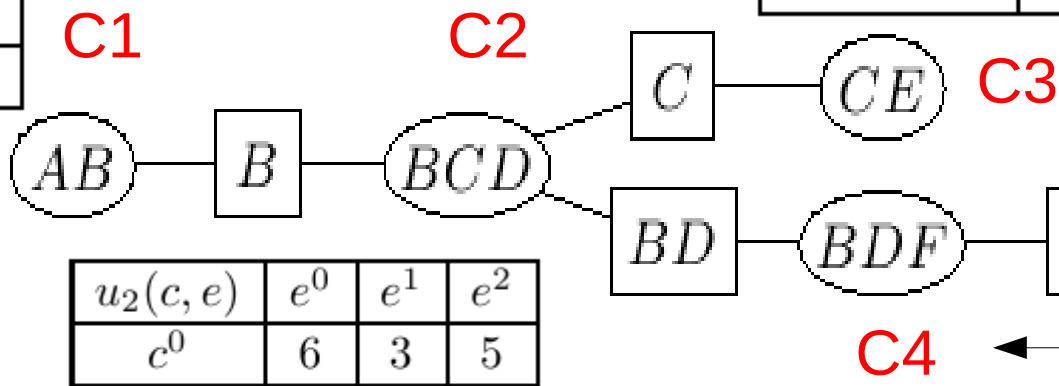
$u_4(b^1,d,f)$	f^0	f^1
d^0	5	8
d^1	9	0

Passagem de um fluxo...

$u_1(a,b)$	b^0	b^1
a^0	8	2
a^1	4	3
a^2	1	7

$u_3(b^0,c,d)$	d^0	d^1
c^0	0	2
c^1	5	1

$u_3(b^1,c,d)$	d^0	d^1
c^0	7	1
c^1	2	4



$u_5(b,g)$	g^0	g^1
b^0	0	9
b^1	6	4

$u_2(c,e)$	e^0	e^1	e^2
c^0	6	3	5
c^1	3	4	0

$u_4(b^0,d,f)$	f^0	f^1
d^0	4	2
d^1	3	8

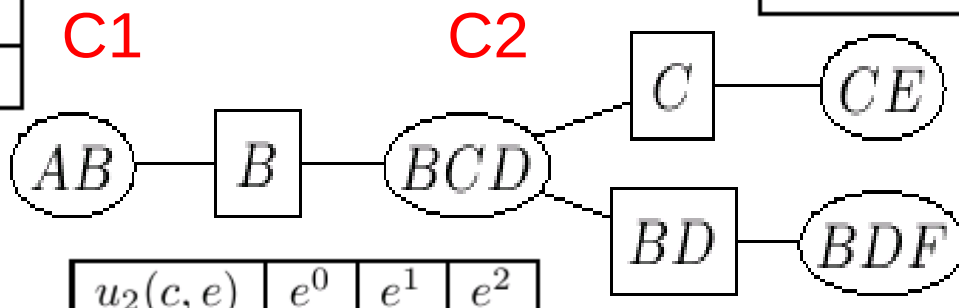
$u_4(b^1,d,f)$	f^0	f^1
d^0	5	8
d^1	9	0

Passando um fluxo...

$u_1(a,b)$	b^0	b^1
a^0	8	2
a^1	4	3
a^2	1	7

$u_3(b^0,c,d)$	d^0	d^1
c^0	0	2
c^1	5	1

$u_3(b^1,c,d)$	d^0	d^1
c^0	7	1
c^1	2	4



$u_2(c,e)$	e^0	e^1	e^2
c^0	6	3	5
c^1	3	4	0

$u_5(b,g)$	g^0	g^1
b^0	0	9
b^1	6	4

C4

$u_4(b^0,d,f)$	f^0	f^1
d^0	4	2
d^1	3	8

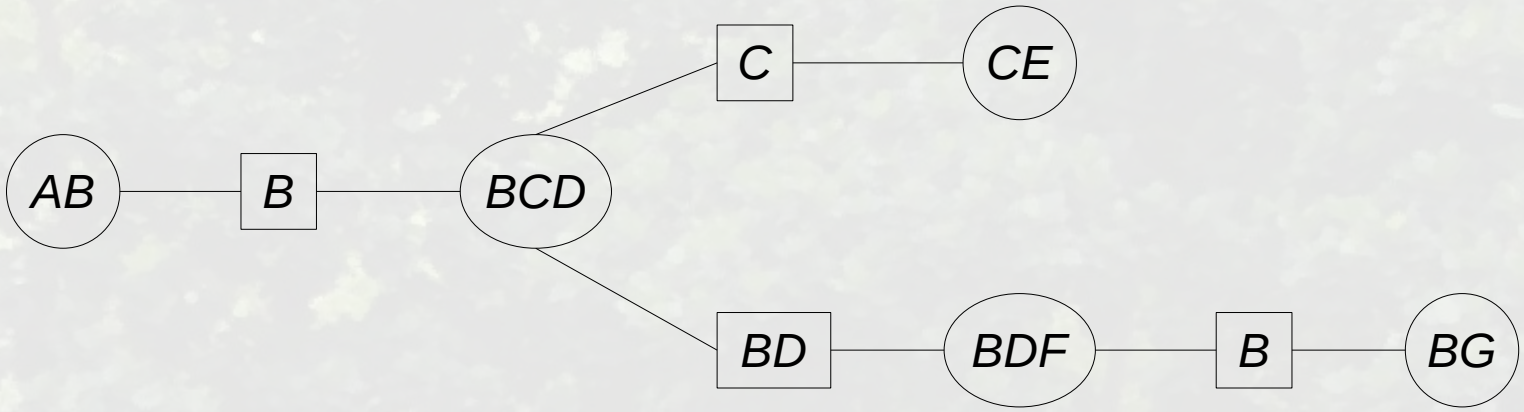
$u_4(b^1,d,f)$	f^0	f^1
d^0	5	8
d^1	9	0

+ +

Plano

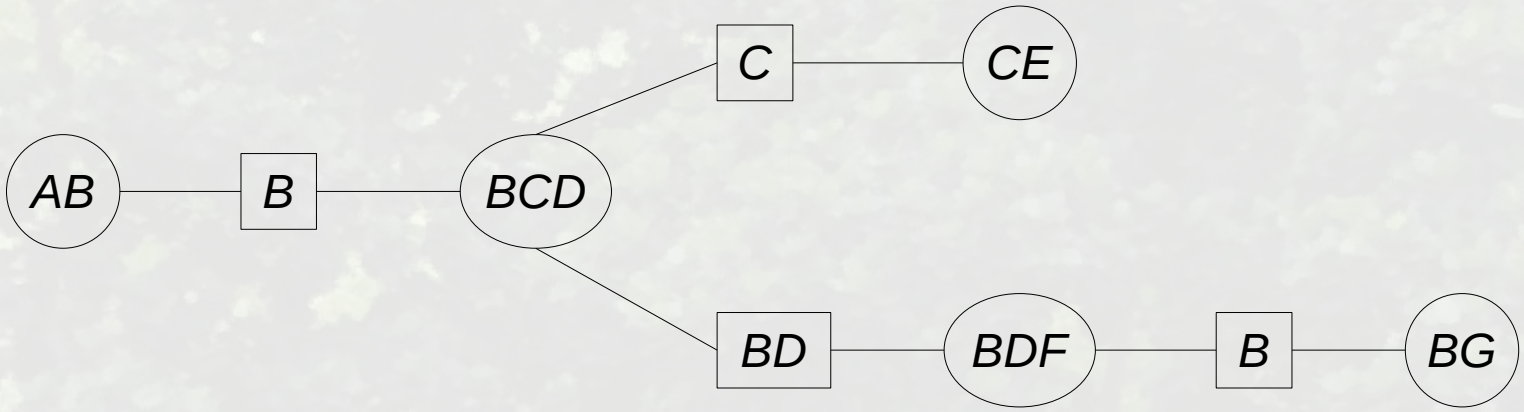
- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Ranking



- Configuração ótima: (*a'*, *b'*, *c'*, *d'*, *e'*, *f'*, *g'*)
- Qual é a próxima melhor configuração?
  - Ela difere de (*a'*, *b'*, *c'*, *d'*, *e'*, *f'*, *g'*) no valor de pelo menos um atributo
- Estratégia: dividir para conquistar
  - Fatiar o conjunto de alternativas restantes em uma partição de conjuntos fáceis de explorar.

Ranking



- E1: (B, C, D) ≠ (b', c', d')
- E2: (B, C, D) = (b', c', d') et (B, D, F) ≠ (b', d', f')
- E3: (B, C, D, F) = (b', c', d', f') et (B, G) ≠ (b', g')
- E4: (B, C, D, F, G) = (b', c', d', f', g') et (C, E) ≠ (c', e')
- E5: (B, C, D, E, F, G) = (b', c', d', e', f', g') et (A, B) ≠ (a', b')

Algoritmo: Utilizar o algoritmo de otimização em cada conjunto e escolher a tupla de maior utilidade encontrada

Ranking: integração de restrições

- Restrições com poucas variáveis
  - Diretamente como termos de utilidade com valor de utilidade $-\infty$ para as configurações proibidas e 0 para as outras
- Restrições que envolvem “muitas variáveis”
  - Exemple: knapsack 0-1

Maximizar $u(x_1,\ldots,x_n)=\sum_{i=1}^{k} u_i(x_{C_i})$

sob a restrição que $\sum_{j=1}^{n} w_j x_j \leqslant c,$

Função GAI decomponível

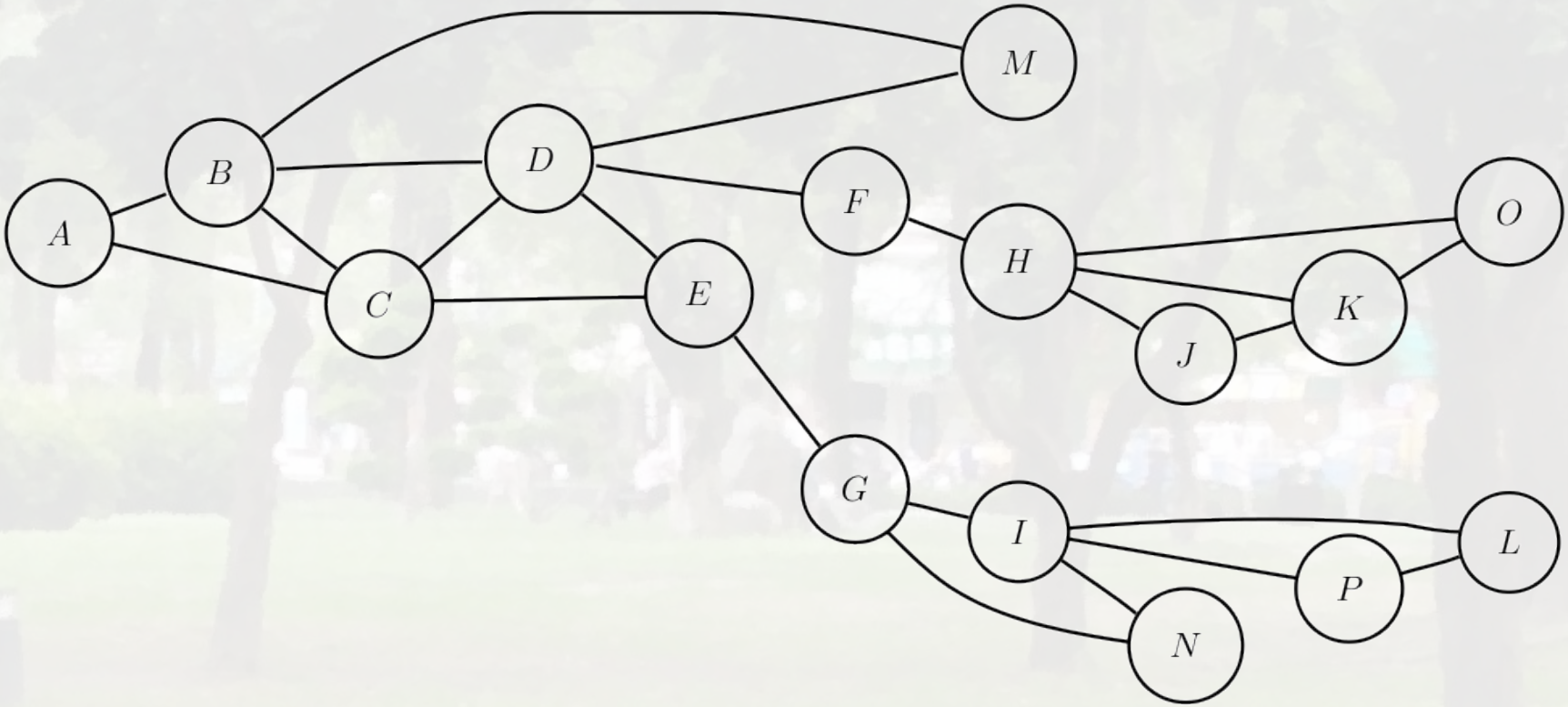
  - Possibilidade de adaptar o algoritmo de ranking e o utilizar num algoritmo do tipo branch-and-bound

Restrição a árvores?

- Os métodos previamente mencionados se baseiam em árvores de junção
- Isso significa que somente funções GAI com termos de utilidade que possam ser representados diretamente por árvores de junção podem ser tratados?
  - Não, grafos quaisquer podem ser recompilados em árvores de junção
  - Exemplo (Jensen et al. 1994):
    - $u(a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p) = u_1(a, b, c) + u_2(b, c, d) + u_3(c, d, e) + u_4(b, d, m) + u_5(f, d) + u_6(f, h) + u_7(h, j, k) + u_8(h, k, o) + u_9(e, g) + u_{10}(g, i, n) + u_{11}(i, l, p)$

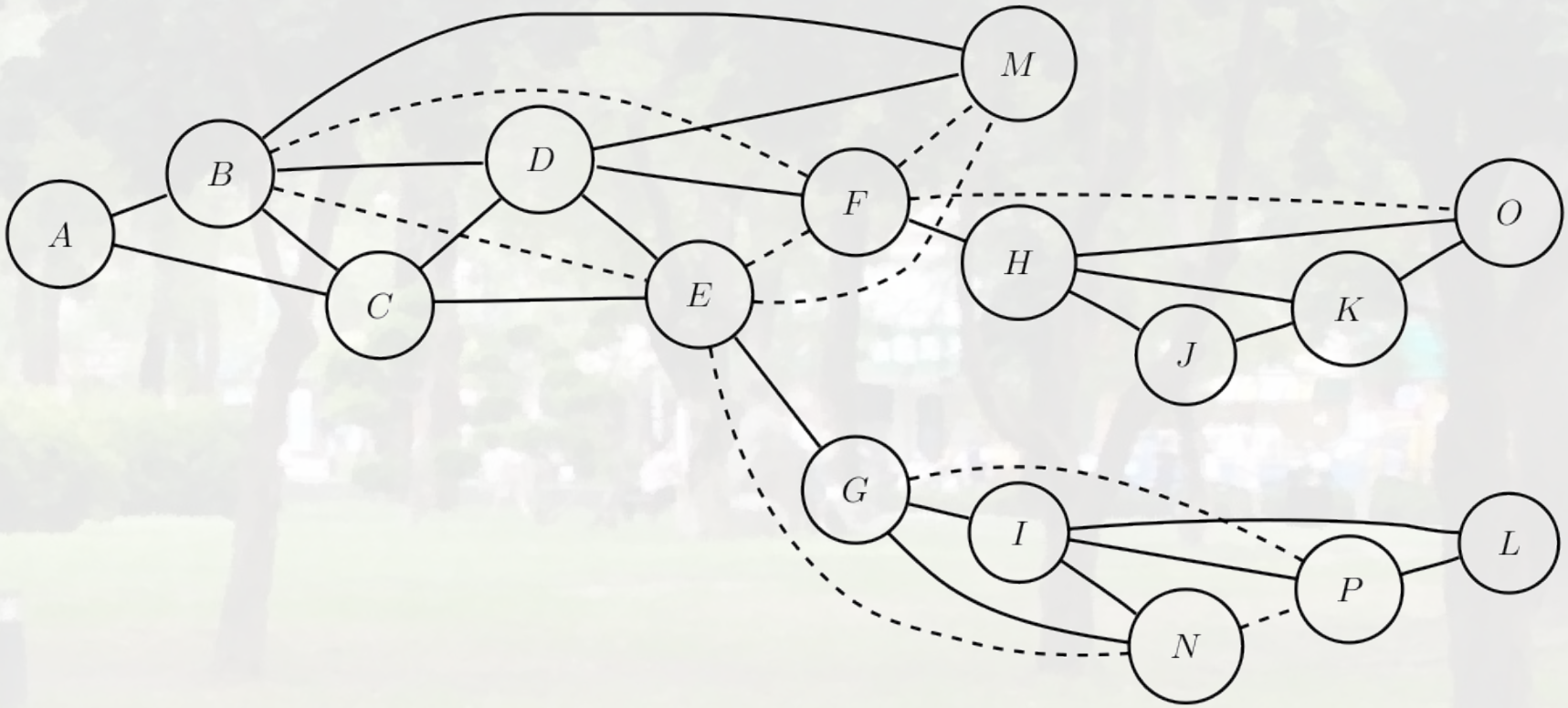
Restrição a árvores?

- $u(a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p) = u_1(a, b, c) + u_2(b, c, d) + u_3(c, d, e) + u_4(b, d, m) + u_5(f, d) + u_6(f, h) + u_7(h, j, k) + u_8(h, k, o) + u_9(e, g) + u_{10}(g, i, n) + u_{11}(i, l, p)$
- 1º passo: grafo de Markov
  - Grafo não-direcionado, com uma aresta entre todo par de atributos que aparecem num mesmo termo de utilidade



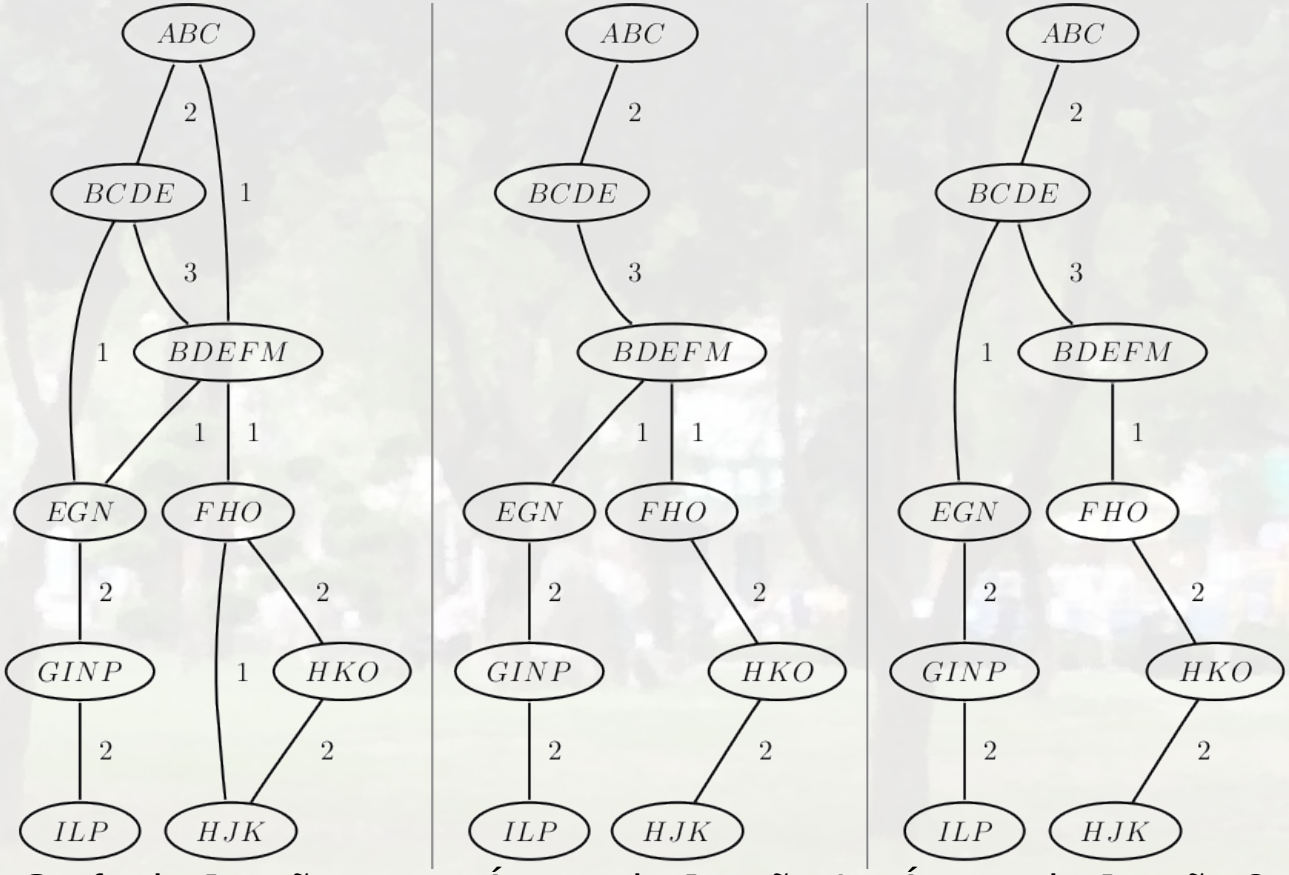
Restrição a árvores?

- $u(a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p) = u_1(a, b, c) + u_2(b, c, d) + u_3(c, d, e) + u_4(b, d, m) + u_5(f, d) + u_6(f, h) + u_7(h, j, k) + u_8(h, k, o) + u_9(e, g) + u_{10}(g, i, n) + u_{11}(i, l, p)$
- 2º passo: grafo de Markov
  - Triangular o grafo (ciclo de tamanho ≥ 4 possui uma corda)



Restrição a árvores?

- $u(a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p) = u_1(a, b, c) + u_2(b, c, d) + u_3(c, d, e) + u_4(b, d, m) + u_5(f, d) + u_6(f, h) + u_7(h, j, k) + u_8(h, k, o) + u_9(e, g) + u_{10}(g, i, n) + u_{11}(i, l, p)$
- 3º passo: grafo de junção (cliques de tamanho máximo)
- 4º passo: árvores de junção (spanning tree de custo máx.)



Grafo de Junção | Árvore de Junção 1 | Árvore de Junção 2

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Agregação

- $(u^1, ..., u^m)$ funções de utilidade GAI em $X = X_1 \times X_2 \times ... \times X_n$
- Algumas interpretações possíveis:
  - **Decisão coletiva** : u^i representa as preferências de um indivíduo *i* em um problema de decisão coletiva
  - **Decisão multi-critério** : u^i representa um ponto de vista segundo um critério dado em um problema multi-critério (por exemplo: beleza, conforto, segurança)

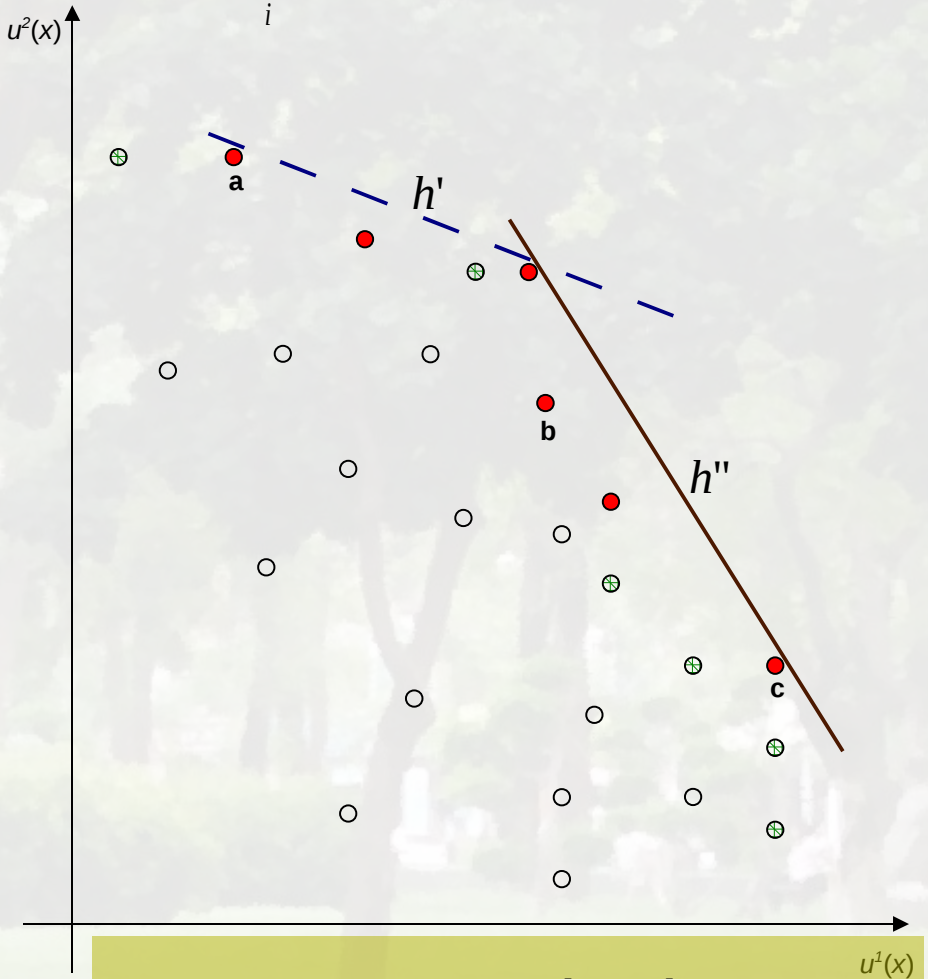
Problema: Nós buscamos a melhor solução de compromisso em X, obtida pela otimização de uma função de agregação

$$f(x) = h(u^1(x), ..., u^m(x))$$

Que funçao de agregação? Exemplos: soma ponderada, Tchebycheff (NP-Hard)

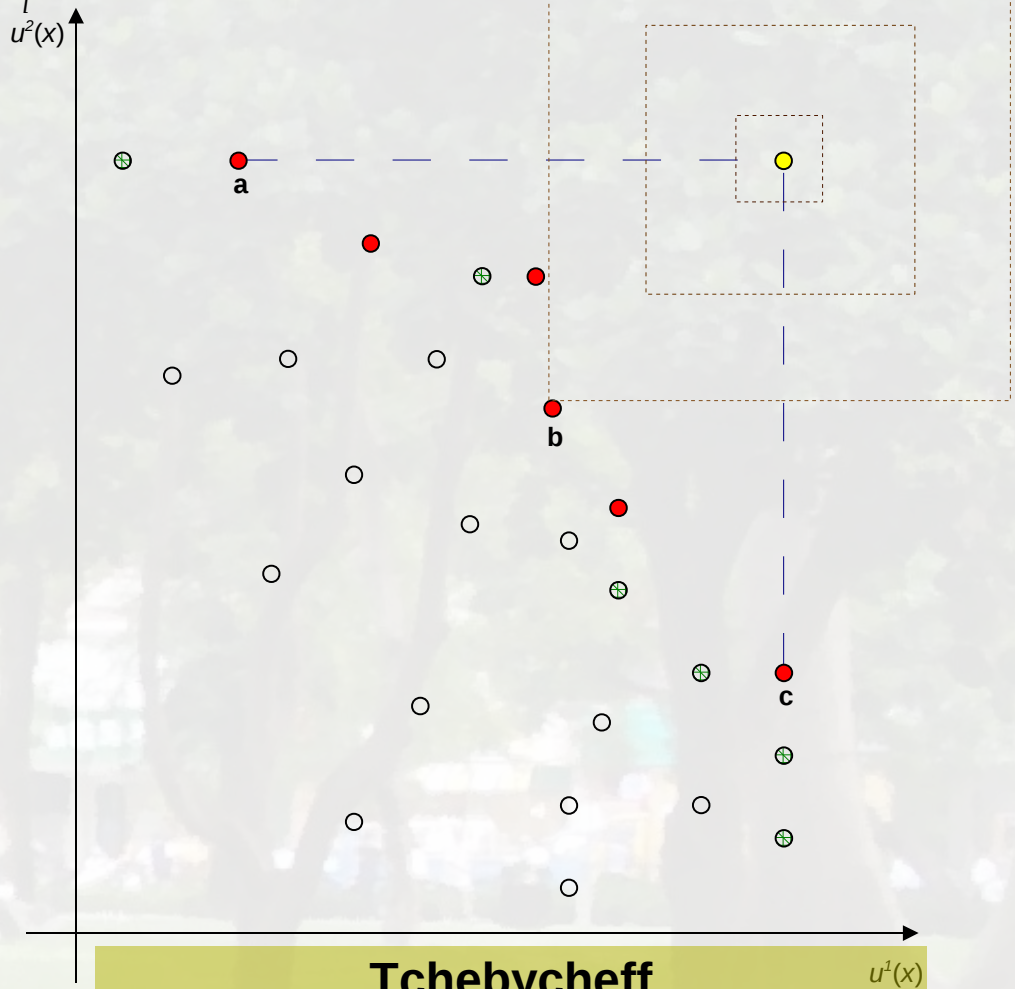
Agregação: funções

$$f_\omega(x) = \sum_i \omega_i u^i(x)$$



Soma ponderada
+ GAI décomponível
- soluções no envelope convexo

$$f_\omega(x) = \max_i \left\{ \omega_i \left| u^i(x) - u^i* \right| \right\}$$



Tchebycheff
- não é GAI decomponível
+ pode atingir todas as soluções Pareto-ótimas

Agregação: estratégia de maximização de *f*
(Gonzales, Perny, Queiroz 08)

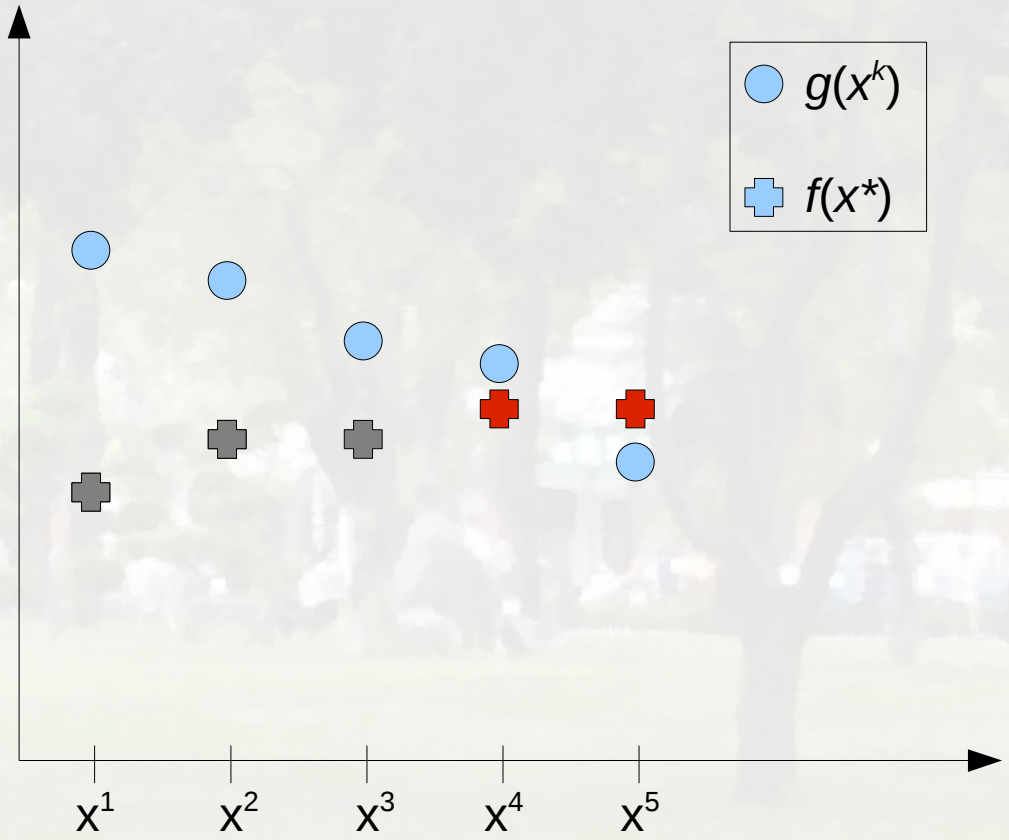
(1) Reformulação : escolher uma função GAI-decomponível $g(x)$ tal que $f(x) \preceq g(x)$ para todo $x \in X$

(2) Ranking : enumerar as soluções de X na ordem decrescente de utilidade segundo g(x).

(3)Critério de parada : parar a enumeração desde que possamos garantir que encontramos a configuração que maximiza *f*

Agregação: critério de parada

- Proposição
  - Sejam x^1, ..., x^k as k-melhores soluções segundo a função GAI g. Se $g(x^k) \leq f(x^*)$ para um k, com $x^* = \text{Argmax}_{i = 1, ..., k} f(x^i)$, então x* é maximal para f, isto é $f(x^*) = \max_{x \in X} f(x)$



Escolha de *g*

- Se *g* é uma combinação linear de funções de utilidade GAI (u^1, ..., u^m), ela é também uma função de utilidade GAI
- Exemplos de funções *f* e *g*

Critères	$f(x)$	$g(x)$
max-min	$\min_j(u^j(x))$	$\frac{1}{m}\sum_j u^j(x)$
min-max regret	$-\max_j(r^j(x))$	$-\frac{1}{m}\sum_j r_j(x)$
Tchebycheff	$-\max_j(w_j r^j(x))$	$-\frac{1}{m}\sum_j w_j r_j(x)$

Onde $r^j(x) = |u^{j*} - u^j(x)|$. Arrependimento do agente *j* se escolhida a alternativa *x* ao invés de u^{j*}

Plano

- Motivação
- Modelos GAI
- Problema de otimização
- Problema de ranking
- Agregação de preferências
- Resumo

Resumo

- Preferências em domínios combinatórios
- Independência e independência condicional
- Utilização de funções de utilidade decomponíveis: aditiva, aditiva generalizada (GAI)
- Modelos GAI para a decisão individual e coletiva
  - Tarefas típicas de um sistema de suporte à decisão
    - otimização e ranking
      - Presença de restrições
    - agregação (não linear)

Bibliografia

- C. Gonzales, P. Perny et S. Queiroz (2008) "GAI-Networks: Optimization, Ranking and Collective Choice in Combinatorial Domains". Foundations of computing and decision sciences, Vol 32, N°4, pp. 3—24.
- C. Gonzales, P. Perny et S. Queiroz (2008) "Preference Aggregation with Graphical Utility Models". In Proceedings of the 23rd AAAI conference on Artificial Intelligence, pp. 1037—1042.
  Download: http://www-desir.lip6.fr/publications/pub_1050_1_AAAI08.pdf
- P. Perny, O. Spanjaard et L.-X. Storme (2006) "A decision-theoretic approach to robust optimization in multivalued graphs". Annals of Operations Research, Vol 147, N°1, pp. 317—341.
  Download: http://www-desir.lip6.fr/publications/pub_402_1_aor.pdf
- C. Gonzales et P. Perny (2005) "GAI Networks for Decision Making under Certainty". In Brafman, R and Junker, U (eds), Proceedings of the 19th International Joint Conference on Artificial Intelligence -- workshop on advances in preference handling, pp. 100—105.
  Download: http://www-desir.lip6.fr/publications/pub_379_1_ijcai05.ps
- C. Gonzales et P. Perny (2004) "GAI Networks for Utility Elicitation". In Proceedings of the 9th International Conference on the Principles of Knowledge Representation and Reasoning, pp. 224—234.
  Download: http://www-desir.lip6.fr/publications/pub_282_1_kr04.ps
- Nilsson, D. 1998. An efficient algorithm for finding the M most probable configurationsin probabilistic expert systems. Statistics and Computing 8, 2 (Jun. 1998), 159-173. DOI= http://dx.doi.org/10.1023/A:1008990218483
- Kjærulff, U. et A/s, J. D. (1990). Triangulation of graphs - algorithms giving small total state space.
- Shibata, Y. (1988). On the tree representation of chordal graphs. Journal of Graph Theory, 12(3):421–428.
- Jensen, F. V. et Jensen, F. (1994). Optimal junction trees. In de Mántaras, R. L. et Poole, D., editors : UAI ’94 : Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence, July 29-31, 1994, Seattle, Washington, USA, pages 360–366. Morgan Kaufmann.