Modeling and Reasoning with Bayesian Networks: Inference by Variable Elimination 6.6-6.9

Relatório semana 5 - MAC0215 (Atividade Curricular em Pesquisa)
Aluno: Renato Lui Geh (Bacharelado em Ciência da Computação)
Orientador: Denis Deratani Mauá

1 Atividades realizadas na semana

Durante a semana foram lidos os seguintes tópicos do livro *Modeling and Reasoning with Bayesian Networks*[1]:

6 - Inference by Variable Elimination

- **6.6** - Choosing an Elimination Order
- **6.7** - Computing Posterior Marginals
- **6.8** - Network Structure and Complexity
- **6.9** - Query Structure and Complexity

2 Definição das atividades

Foram estudados como encontrar a melhor ordem de eliminação de variáveis[3], definimos o que são grafos de interação de Redes Bayesianas, como calcularmos marginais posteriores (*posterior marginals*), algumas propriedades da estrutura de uma rede e como podar (*prune*) nós e arestas de uma Rede Bayesiana.

Esta seção será dividida em subseções onde veremos os itens enumerados no parágrafo acima. A cada tópico que não tenha sido apresentado ainda vamos abordar a teoria por trás e introduzir o assunto. Separaremos tais explicações em uma subsubseção equivalente. Abaixo está a lista de tópicos que iremos abordar.

1. Escolhendo uma ordem de eliminação
   1.1. Subgraphs, induced subgraphs, cliques e spanning subgraphs
   1.2. Grafo de interação
2. Computando posterior marginals
   2.1. Posterior marginals e joint marginals
   2.2. Eliminação de variáveis e posterior marginals
3. Estrutura e complexidade da rede
   3.1. Treewidth
   3.2. Tree networks, polytrees e multiply connected
4. Pruning
   4.1. Pruning nodes
   4.2. Pruning edges
   4.3. Network pruning
5. Apêndice
   5.1. P e NP
   5.2. NP-hard e NP-complete

2.1 Escolhendo uma ordem de eliminação

Como vimos no relatório anterior[3], queremos criar factors que sejam os menores possíveis. A ordem de eliminação que tiver menor w será a melhor. Dizemos que o w de uma ordem de eliminação é o maior *width* de todos os factors da ordem de eliminação.

Antes de acharmos a melhor ordem de eliminação precisamos saber como calcular o w de uma ordem. Um jeito ingênuo de acharmos é, a cada interação i de uma multiplicação e soma, acharmos o w_i com respeito a $\pi(i)$. No final retornamos o maior w_i.

i	$\pi(i)$	$\mathcal{S}$	f_i	w
		$\Theta_A\Theta_{B\|A}\Theta_{C\|A}\Theta_{D\|BC}\Theta_{E\|C}$		
1	B	$\Theta_A\Theta_{C\|A}\Theta_{E\|C}f_1(A,C,D)$	$f_1 = \sum_B \Theta_{B\|A}\Theta_{D\|BC}$	3
2	C	$\Theta_A f_2(A,D,E)$	$f_2 = \sum_C \Theta_{C\|A}\Theta_{E\|C}f_1(A,C,D)$	3
3	A	$f_3(D,E)$	$f_3 = \sum_A \Theta_A f_2(A,D,E)$	2
4	D	$f_4(E)$	$f_4 = \sum_D f_3(D,E)$	1

Na tabela acima a segunda coluna indica a variável $\pi(i)$ que queremos eliminar. A terceira coluna é o resultado da substituição de $\pi(i)$ pelo factor resultante na quarta coluna, onde multiplicamos e somamos as CPTs onde $\pi(i)$ está presente. A última coluna é o w_i de cada eliminação. Pode-se ver que o w dessa ordem é 3, já que é o maior w entre todas as iterações. Note que não precisamos realmente computar a eliminação das variáveis. De fato não fazemos tais eliminações, mas a cada $\pi(i)$ que aparecem nos factors $f(\mathbf{X}_k)$ substituímos tais factors por um novo factor sob as variáveis $\cup_k \mathbf{X}_k \setminus \pi(i)$.

Um outro jeito de se achar a *width* de uma ordem de eliminação é por meio de um grafo não direcionado que representa as interações das CPTs de uma Rede Bayesiana. Antes de definirmos esse grafo vamos introduzir alguns conceitos fundamentais sobre grafos.

2.1.1 Subgraphs, induced subgraphs, cliques e spanning subgraphs

Vamos definir $V(G)$ como o conjunto de vértices do grafo G. Similarmente, $E(G)$ é o conjunto de arestas do grafo G. Dizemos que um grafo é não direcionado se para cada vértice x, y em G, tanto xy quanto yx são arestas válidas em G.

Primeiro vamos ver a definição de subgrafo:

Definição. *Um grafo H é um subgrafo de G se $V(H)$ é um subconjunto de $V(G)$ e $E(H)$ é subconjunto de $E(G)$. Dizemos que G é o supergrafo de H se H é subgrafo de G.*

Vamos agora definir o que é um grafo induzido:

Definição. *Um subgrafo H de um grafo G é dito induzido se para cada par de vértices x, y em H, xy é uma aresta de H se e somente se xy é uma aresta em G. Ou seja, H é um subgrafo induzido de G se as arestas em H são exatamente as arestas que aparecem em G sob o mesmo conjunto de vértices. Se $V(H)$ é um subconjunto S de $V(G)$, então H pode ser escrito como $G[S]$ e é dito ser induzido por S.*

Agora iremos definir o que é a completude de um grafo:

Definição. *Dizemos que um grafo G é completo se todo par de vértices distintos em G é conectado por uma aresta única.*

Definimos agora o que é um clique:

Definição. *Um clique é um subconjunto de vértices de um grafo não direcionado tal que o subgrafo induzido dele seja completo. Ou seja, todo par distinto de vértices em um clique é adjacente.*

Vamos também definir o que é um *spanning subgraph*:

Definição. *Dizemos que um subgrafo H cobre (spans) um grafo G se ele tem mesmo conjunto de vértices que G. H é então dito o spanning subgraph de G.*

2.1.2 Grafo de interação

Agora que temos uma noção básica sobre grafos podemos definir o grafo de interação.

Definição. *Seja $f_1, ..., f_n$ o conjunto de factors. O grafo de interação G desses factors é um grafo não direcionado construído de tal forma que os nós de G são as variáveis que aparecem nos factors $f_1, ..., f_n$. Há uma aresta entre duas variáveis em G se e somente se essas variáveis aparecem no mesmo factor.*

Em outras palavras, as variáveis $\mathbf{X}_i$ de cada factor f_i formam um clique em G, ou seja, as variáveis emparelhadas são adjacentes.

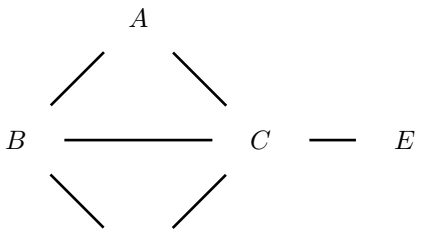
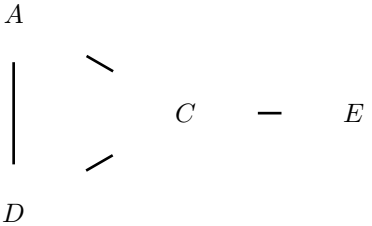
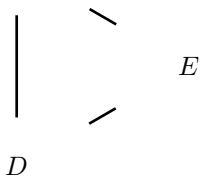
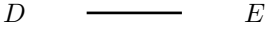
$\mathcal{S}_1 : \Theta_A \Theta_{B\|A} \Theta_{C\|A} \Theta_{D\|BC} \Theta_{E\|C}$	
$\mathcal{S}_2 : \Theta_A \Theta_{C\|A} \Theta_{E\|C} f_1(A, C, D)$	
$\mathcal{S}_3 : \Theta_A f_2(A, D, E)$	
$\mathcal{S}_4 : f_3(D, E)$	
$\mathcal{S}_5 : f_4(E)$	

Tabela 1 Grafo de interação resultante da eliminação das variáveis B, C, A, D nessa ordem.

A Tabela 1 mostra como fica o grafo de interação ao eliminarmos as variáveis B, C, A, D nessa ordem. É importante notar que:

1. Se G é o grafo de interação dos factors $\mathcal{S}$, então eliminar a variável $\pi(i)$ de $\mathcal{S}$ leva a construção de um factor sob os vizinhos de $\pi(i)$ em G. Por exemplo, eliminar a variável B do factor $\mathcal{S}_1$ da Tabela 1 gera um factor sob as variáveis A, C, D, que são os vizinhos de B no grafo de interação.

2. Seja $\mathcal{S}'$ os factors que resultam da eliminação da variável $\pi(i)$ dos factors $\mathcal{S}$. Se G' e G são os grafos de interação de $\mathcal{S}'$ e $\mathcal{S}$ respectivamente, então G' pode ser obtido de G da seguinte forma:

   (a) Adicione uma aresta em G entre cada par de vizinhos da variável $\pi(i)$ que não estejam já conectados por uma aresta.

   (b) Remova a variável $\pi(i)$ de G.

Pode-se ver que (a) corresponde a multiplicar os factors que contém a variável $\pi(i)$ em $\mathcal{S}$ e (b) equivale a somar a variável $\pi(i)$ do factor resultante.

Computar a *width* de uma ordem é útil quando temos um número pequeno de ordens. No entanto, quando temos um número muito grande de ordens precisamos achar uma ordem ótima de um jeito melhor. Achar uma ordem ótima é NP-difícil (*NP-hard*), mas podemos usar várias heurísticas que resultam em boas ordens.

Uma das heurísticas mais populares é a *min-degree*: sempre elimine a variável que acarreta na construção do menor factor possível. Se usarmos o grafo de interação podemos ver que a heurística diz que devemos eliminar a variável que tem o menor número de vizinhos no grafo de interação atual. A min-degree é ótima para redes que possuem uma ordem de eliminação de $width \leq 2$.

Outra heurística que na maioria das vezes é mais efetiva que a min-degree é a *min-fill*, onde nós eliminamos a variável de tal forma que precisemos adicionar o menor número possível de arestas no grafo de interação.

2.2 Computando posterior marginals

Vamos mostrar agora como computar posterior marginals a partir de eliminação de variáveis. Mas primeiro vamos definir o que é posterior marginal e joint marginals.

2.2.1 Posterior marginals e joint marginals

Dada uma distribuição de probabilidade conjunta[2] $Pr(x_1, ..., x_n)$ a distribuição marginal $Pr(x_1, ..., x_m), m \leq n$ é definida como:

$$Pr(x_1, ..., x_m) = \sum_{x_{m+1}, ..., x_n} Pr(x_1, ..., x_n) \tag{1}$$

Ou seja, a distribuição marginal pode ser vista como uma *projeção* da distribuição conjunta num conjunto menor $X_1, ..., X_m$.

Quando a distribuição marginal é computada dada uma evidência $\mathbf{e}$,

$$Pr(x_1, ..., x_m|\mathbf{e}) = \sum_{x_{m+1}, ..., x_n} Pr(x_1, ..., x_n|\mathbf{e}), \tag{2}$$

Então dizemos que ela é uma *posterior marginal*. Quando uma distribuição marginal não é dada nenhuma evidência, então dizemos que ela é uma *prior marginal*.

Vamos agora definir uma marginal conjunta (*joint marginal*). Uma joint marginal é uma variação da posterior marginal da forma $P(\mathbf{Q}, \mathbf{e})$. Ou seja, ao invés de computarmos a probabilidade de $\mathbf{q}$ dado $\mathbf{e}$, $Pr(\mathbf{q}|\mathbf{e})$, nós computamos a probabilidade de $\mathbf{q}$ e $\mathbf{e}$, $Pr(\mathbf{q}, \mathbf{e})$. Por exemplo, tomemos as Tabelas 2 e 3 dados $\mathbf{Q} = \{D, E\}$ e $\mathbf{e} : A = true, B = false$ na mesma rede da Tabela 1:

D	E	$Pr(\mathbf{Q}\|\mathbf{e})$
true	true	0.448
true	false	0.192
false	true	0.112
false	false	0.248

D	E	$Pr(\mathbf{Q}, \mathbf{e})$
true	true	0.21504
true	false	0.09216
false	true	0.05376
false	false	0.11904

Tabelas 2 e 3 Posterior marginal e joint marginal respectivamente.

A terceira linha da Tabela 3 (joint marginal) diz que:

$$Pr(D = false, E = true, A = true, B = false) = 0.05376 \tag{3}$$

Ao somarmos todas as probabilidades da joint marginal nesse factor, teremos como resultado 0.48, que é a probabilidade de evidência $\mathbf{e} : A = true, B = false$. Se somarmos todas as probabilidades que aparecem na joint marginal, teremos sempre a probabilidade da evidência $\mathbf{e}$. Isso quer dizer que podemos computar a posterior marginal $Pr(\mathbf{Q}|\mathbf{e})$ normalizando a joint marginal $Pr(\mathbf{Q}, \mathbf{e})$.

2.2.2 Eliminação de variáveis e posterior marginals

Podemos usar eliminação de variáveis para computar joint marginals se zerarmos todas as linhas na distribuição de probabilidade conjunta onde há uma inconsistência[3] com a evidência $\mathbf{e}$. Mais formalmente dizemos que:

Definição. *A redução de um factor $f(\boldsymbol{X})$ dado uma evidência e é um outro factor sob as variáveis $\boldsymbol{X}$ denotado por f^e e definido como:*

$$f^e(\boldsymbol{x}) \stackrel{\text{def}}{=} \begin{cases} f(\boldsymbol{x}), & se\ \boldsymbol{x} \sim \boldsymbol{e} \\ 0, & caso\ contrário. \end{cases} \tag{4}$$

Portanto, dado o factor da Tabela 2 e evidência $\mathbf{e} : E = true$, teríamos:

D	E	$f^{\mathbf{e}}$
true	true	0.448
true	false	0
false	true	0.112
false	false	0

Para facilitar omitimos todas as linhas zeradas:

D	E	$f^{\mathbf{e}}$
true	true	0.448
false	true	0.112

Dado $\mathbf{Q} = \{D, E\}$ e $\mathbf{e} : A = true, B = false$, a marginal conjunta $Pr(\mathbf{Q}, \mathbf{e})$ pode ser computada como:

$$Pr(\mathbf{Q}, \mathbf{e}) = \sum_{A,B,C} (\Theta_{E|C}\Theta_{D|BC}\Theta_{C|A}\Theta_{B|A}\Theta_{A})^{\mathbf{e}} \tag{5}$$

Como pode-se ver, ainda temos o problema da complexidade, já que precisamos multiplicar todas as CPTs para depois eliminar as variáveis. No entanto, podemos aplicar o seguinte teorema:

Teorema. *Se f_1 e f_2 são dois factors e $\boldsymbol{e}$ é uma instância, então:*

$$(f_1 f_2)^{\boldsymbol{e}} = f_1^{\boldsymbol{e}} f_2^{\boldsymbol{e}} \tag{6}$$

Portanto podemos reduzir para:

$$Pr(\mathbf{Q} = \{D, E\}, \mathbf{e}) = \sum_{A,B,C} \Theta^{\mathbf{e}}_{E|C}\Theta^{\mathbf{e}}_{D|BC}\Theta^{\mathbf{e}}_{C|A}\Theta^{\mathbf{e}}_{B|A}\Theta^{\mathbf{e}}_{A} \tag{7}$$

Essa forma mantém a distribuição de probabilidade conjunta em forma fatorada e portanto permite eliminarmos as variáveis nas CPTs reduzidas.

Para calcularmos a posterior marginal $Pr(C|A = true)$, dados:

$$\begin{aligned} Pr(C = true, A = true) &= 0.192 \\ Pr(C = false, A = true) &= 0.408 \\ Pr(A = true) &= 0.600 \end{aligned} \tag{8}$$

Precisamos normalizar o factor, ou seja:

C	$Pr(C\|A = true)$
true	0.32
false	0.68

Portanto $Pr(C = true|A = true) = 0.32$ e $Pr(C = false|A = true) = 0.68$.

Note que se tentarmos calcular $Pr(\mathbf{Q}|\mathbf{e})$ com $\mathbf{Q}$ vazio, estaremos calculando um factor trivial com valor igual a probabilidade de evidência $\mathbf{e}$.

2.3 Estrutura e complexidade da rede

Apesar de termos duas Redes Bayesianas com o mesmo número de variáveis, as ordens de eliminação ótimas de cada uma podem ser diferentes entre si. Isso por causa da *treewidth* das redes. Vamos definir o que é uma treewidth, falar sobre NP-completude e NP-dificuldade e em seguida classificar as várias estruturas que uma rede pode ter.

2.3.1 Treewidth

Definimos como treewidth o número que quantifica o quanto uma rede se assemelha à estrutura de uma árvore. Em particular, quanto mais similar a uma árvore, menor o seu treewidth.

Um ponto importante sobre treewidth é que nao há nenhuma ordem de eliminação completa onde sua *width* é menor que a treewidth da rede. Adicionalmente, existe uma ordem de eliminação cuja *width* é igual à treewidth da rede e ela é ótima. No entanto, determinar tal ordem é NP-difícil. Discutiremos mais sobre isso no Apêndice deste relatório.

Vamos descrever algumas características da treewidth:

- O número de nós não tem efeito na treewidth.
- O número de pais por nó tem efeito direto na treewidth. Se um número de pais por nó chega k, então o treewidth não é menor que k.
- Loops tendem a aumentar a treewidth.
- O número de loops não tem efeito na treewidth. As interações geradas pelo loop tem efeito direto na treewidth.

Vamos agora falar sobre algumas classes de redes com treewidths conhecidas.

2.3.2 Tree networks, polytrees e multiply connected

Polytree networks, também conhecidas como *singly connected networks* são redes onde há no máximo um caminho (não direcionado) entre qualquer dois nós. A treewidth dessas redes é k, onde k é o número máximo de pais que qualquer nó pode ter.

Tree networks são polytrees onde cada nó tem no máximo um pai, e portanto sua treewidth é no máximo 1.

Multiply connected são aquelas que não são singly connected.

2.4 Pruning

Queremos aplicar transformações na estrutura da rede de tal forma que melhoramos como achamos inferência e ao mesmo tempo preservando a joint marginal $Pr(\mathbf{Q}, \mathbf{e})$ e portanto a prior e posterior marginals.

Uma dessas transformações é chamada de *pruning* (corte). Vamos nos referir à ação de pruning (*to prune*) como *aparar*. Quando nos referimos ao substantivo pruning vamos usar a tradução *corte*.

2.4.1 Pruning nodes

Dada uma Rede Bayesiana $\mathcal{N}$ e query $(\mathbf{Q}, \mathbf{e})$, podemos remover qualquer nó folha $\mathcal{L}$ (junto com sua CPT) desde que $\mathcal{L} \notin (\mathbf{Q} \cup \mathbf{E})$ e não afete a habilidade da rede de computar a query corretamente. Podemos aplicar essa transformação iterativamente, aparando vários nós. Denominamos o resultado de se remover as folhas nós do jeito citado como $pruneNodes(\mathcal{N}, \mathbf{Q} \cup \mathbf{E})$.

Teorema. *Seja uma Rede Bayesiana $\mathcal{N}$ e uma query $(\mathbf{Q}, \mathbf{e})$. Se $\mathcal{N}' = pruneNodes(\mathcal{N}, \mathbf{Q} \cup \mathbf{E})$, então $Pr(\mathbf{Q}, \mathbf{e}) = Pr'(\mathbf{Q}, \mathbf{e})$, onde Pr e Pr' são as distribuições de probabilidade induzidas pelas redes $\mathcal{N}$ e $\mathcal{N}'$ respectivamente.*

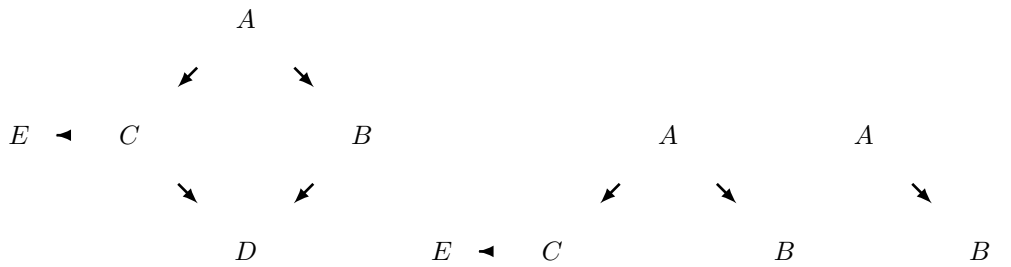


Tabela 4 Aparando nós na rede dados duas queries diferentes.

A Tabela 4 mostra a rede com dois cortes. No primeiro queremos a marginal sob B e E sem nenhuma evidência. Portanto aparamos a folha D. Depois deste corte não podemos aparar nenhuma outra folha nó, já que todas aparecem na query. Na segunda queremos a marginal sob a variável B sem evidência. Aparamos D e E, mas assim fazemos este corte podemos em seguida aparar o nó C, já que ele não aparece na query. Note que a primeira figura da Tabela 4 tem treewidth 2, enquanto que as treewidth da segunda e terceira figuras são 1.

Se as variáveis $\mathbf{Q} \cup \mathbf{E}$ forem próximas das raízes da rede então elas reduzem significantemente a treewidth da rede. No pior dos casos todas as folhas aparecem em $\mathbf{Q} \cup \mathbf{E}$ e portanto nenhum corte é possível. No melhor dos casos as variáveis $\mathbf{Q} \cup \mathbf{E}$ contêm somente nós raízes, permitindo o corte de todos os nós exceto os que pertencem a $\mathbf{Q} \cup \mathbf{E}$.

2.4.2 Pruning edges

Dada uma Rede Bayesiana $\mathcal{N}$ e uma query $(\mathbf{Q}, \mathbf{e})$, podemos eliminar uma aresta da rede e reduzir algumas CPTs sem alterar a habilidade de computar a joint marginal $Pr(\mathbf{Q}, \mathbf{e})$ corretamente. Em particular, para cada aresta $U \rightarrow X$ que origina do nó U em $\mathbf{E}$, podemos:

1. Remover a aresta $U \rightarrow X$ da rede.
2. Substituir a CPT $\Theta_{X|\mathbf{U}}$ do nó X por uma CPT menor obtida de $\Theta_{X|\mathbf{U}}$ assumindo valor de u do nó pai U dado uma evidência $\mathbf{e}$. Essa nova CPT corresponde a $\sum_U \Theta^u_{X|\mathbf{U}}$.

Chamamos essa transformação de $pruneEdges(\mathcal{N}, \mathbf{e})$.

Teorema. *Seja $\mathcal{N}$ uma Rede Bayesiana e $\mathbf{e}$ uma instância. Se $\mathcal{N} = pruneEdges(\mathcal{N}, \mathbf{e})$, então $Pr(\mathbf{Q}, \mathbf{e}) = Pr'(\mathbf{Q}, \mathbf{e})$, onde Pr e Pr' são as distribuições de probabilidade induzidas pelas redes $\mathcal{N}$ e $\mathcal{N}'$ respectivamente.*

É importante ressaltar que a rede aparada funciona apenas para queries da forma $Pr(\mathbf{q}, C = false)$. Se a instância $C = false$ não aparece na query, as respostas serão diferentes da rede original.

2.4.3 Network pruning

O resultado de se aparar os nós e as arestas de uma rede será chamada de corte de uma árvore (*network pruning*), e dado uma query $(\mathbf{Q}, \mathbf{e})$, o corte de uma árvore será denominada por $pruneNetwork(\mathcal{N}, \mathbf{Q}, \mathbf{e})$. Pode-se dizer que em geral uma network pruning pode trazer uma redução significativa na treewidth da rede.

Definição. *A treewidth efetiva (effective treewidth) de uma Rede Bayesiana $\mathcal{N}$ dada uma query $(\mathbf{Q}, \mathbf{e})$ é a treewidth de $pruneNetwork(\mathcal{N}, \mathbf{Q}, \mathbf{e})$.*

É importante notar que o corte de uma Rede Bayesiana pode ser feita em tempo linear ao tamanho da rede (o tamanho das CPTs dela). Portanto, é geralmente uma boa ideia aparar a Rede Bayesiana antes de respondermos as queries.

2.5 Apêndice

O capítulo 6 do livro[1] requer muito conteúdo que não é visto no próprio livro. Exemplos disso são Teoria de Grafos e assuntos relacionados a NP. Para grafos foi decidido que fossem explicadas algumas definições diretamente no tópico onde as mencionamos. Para NP foi decidido que ficasse em uma seção distinta - o apêndice - por ser um assunto mais amplo que abrange todos os tópicos mencionados.

2.5.1 P e NP

Vamos a seguir definir o que significa o termo NP e P.

NP significa *nondeterministic polynomial time*. NP é o conjunto de todos os problemas de decisão na qual as instâncias onde as respostas são "verdadeiro" tem uma prova eficientemente verificável para o fato que a resposta seja realmente "verdadeira". Mais formalmente:

Definição. *NP é o conjunto de problemas de decisão onde as instâncias verdadeiras são aceitas em tempo polinomial por uma máquina de Turing não-determinística.*

Uma máquina de Turing não-determinística é definida abaixo:

Definição. *Uma máquina de Turing não-determinística pode ter um conjunto de regras onde há mais de uma ação para uma dada situação, contrastando com uma máquina de Turing determinística que pode ter apenas uma ação para uma dada situação.*

NP é uma classe de complexidade. P está contido em NP. P é uma classe de complexidade onde os problemas computacionais são "tratáveis", ou seja, podem ser reduzidos em tempo polinomial.

2.5.2 NP-hard e NP-complete

Vamos agora definir NP-dificuldade (*NP-hardness*) e NP-completude (*NP-completeness*).

NP-dificuldade é uma classe de complexidade onde os problemas contidos são "pelo menos tão difíceis quanto os problemas mais difíceis em NP". Mais formalmente:

Definição. *Um problema H é NP-difícil quando todo problema L em NP pode ser reduzido em tempo polinomial para H.*

Vamos entender o que significa reduzir um problema. Mas para isso precisamos entender o que significa fechamento em relação a uma operação:

Definição. *Seja A um conjunto e uma operação $\circ$. Dizemos que um conjunto é fechado em relação a $\circ$ se para todo pair de elementos a_i e a_j pertencentes a A, $a_i \circ a_j = a_k$, onde $a_k \in A$. Ou seja, todo elemento resultante de uma operação dada com dois elementos de um conjunto gera um elemento que pertence ao mesmo conjunto.*

Antes de vermos a definição de redução, vamos lembrar o que significa composição de função. Chamamos de composição de função a aplicação de uma função com o resultado de uma outra para produzir uma terceira função. Escrevemos a composição de uma função $f : X \to Y$ e $g : Y \to Z$ como $g(f(x))$ ou $g \circ f$.

Finalmente vamos ver a definição de redução:

Definição. *Dados dois subconjuntos A e B em $\boldsymbol{N}$, onde $\boldsymbol{N}$ é fechado em relação a composição, e um conjunto F de N para N, A é chamado de redutível em B sob F se*

$$\exists f \in F \quad \forall x \in \mathbb{N} \quad x \in A \iff f(x) \in B \tag{9}$$

Escrevemos

$$A \leq_F B \tag{10}$$

Seja S um subconjunto de $\boldsymbol{P(N)}$ e $\leq$ uma redução, então S é chamado de fechado sob $\leq$ se

$$\forall s \in S \quad \forall A \in P(N) \quad A \leq s \Rightarrow A \in S \tag{11}$$

Um subconjunto A de $\mathbf{N}$ é chamado de difícil para S se

$$\forall s \in S \quad s \leq A \tag{12}$$

Um subconjunto A de $\mathbf{N}$ é chamado de completo por S se A é difícil para S e A está em S.

Em outras palavras, um problema A é redutível em B se um algoritmo para solucionar o problem B eficientemente (se ele existe) também pode ser usado como uma subrotina para solucionar o problema A eficientemente. Se isso for verdade, então solucionar A não pode ser mais difícil que solucionar B.

Portanto, podemos dizer que um problema de decisão H é NP-difícil quando para qualquer problema L in NP, existe uma redução para tempo polinomial de L para H. Ou seja, existe uma redução para tempo polinomial de um problem NP-completo G para H. Para entendermos essa segunda possível definição, vamos definir NP-completude.

Definição. *Dizemos que um problema é NP-completo quando ele está em NP e NP-difícil. Ou seja, um problema de decisão C é NP-completo se:*

1. *C está em NP e*
2. *Todo problema em NP é redutível para C em tempo polinomial.*

Note que um problema que satisfaça o item 2 é NP-difícil, satisfaça ou não o item 1. Problemas NP-completos são, em NP, o conjunto de todos os problemas de decisão cujas soluções podem ser verificadas em tempo polinomial. Um problema p em NP é NP-completo se todo outro problema em NP pode ser transformado (ou reduzido) para p em tempo polinomial.

Referências

[1] Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. 1st Edition. Cambridge University Press, 2009.

[2] Renato Lui Geh. *Aprendizado Automático de Sum-Product Networks (SPN)*. 2015. URL: http://www.ime.usp.br/~renatolg/mac0215/doc/project/relatorio.pdf.

[3] Renato Lui Geh. *Modeling and Reasoning with Bayesian Networks: Inference by Variable Elimination 6.1-6.5*. 2. 2015. URL: http://www.ime.usp.br/~renatolg/mac0215/doc/reports/week2/relatorio.pdf.