Geração uniforme de *k-trees* para aprendizado de redes bayesianas

Tiago Madeira
<madeira@ime.usp.br>

Supervisor: Prof. Dr. Denis Deratani Mauá

Bacharelado em Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

Novembro de 2016

No que consiste o trabalho?

Estudo sobre amostragem uniforme de *k-trees* e seu uso no aprendizado da estrutura de redes bayesianas com *treewidth* limitado.

Por que estudar *k-trees*?

Há interesse considerável em desenvolver ferramentas eficientes para manipular *k-trees*, porque **problemas NP-difíceis são resolvidos em tempo polinomial** em *k-trees* e subgrafos de *k-trees*.

Alguns exemplos[1]:

- Encontrar tamanho máximo dos conjuntos independentes;
- Computar tamanho mínimo dos conjuntos dominantes;
- Calcular número cromático;
- Determinar se tem um ciclo hamiltoniano.

[1] Stefan Arnborg, Andrzej Proskurowski. Linear time algorithms for NP-Hard problems restricted to partial *k*-trees. *Discrete Applied Mathematics*, 23:11–24, 1989.

Por que gerar *k-trees?*

Há muitas razões, como por exemplo para testar a eficácia de algoritmos aproximados.

O problema que desperta nosso interesse é o **aprendizado de redes bayesianas**.

O que foi feito?

- Implementação do algoritmo de Caminiti *et al.* (2010)[2] para **codificar** ***k*****-trees de forma bijetiva em tempo linear**.
- Implementação de algoritmo para **amostrar** ***k*****-trees uniformemente** e testes para comprovar seu funcionamento.
- Estudo sobre **aprendizado de redes bayesianas com treewidth limitado** por meio da amostragem uniforme de *k-trees* conforme artigo de Nie *et al.* (2014)[3].
- **Comparação entre métodos** para aprender redes bayesianas.

[2] Severio Caminiti, Emanuele G. Fusco, Rossella Petreschi. Bijective linear time coding and decoding for *k*-trees. *Theory of Computing Systems*, 46:284–300, 2010.

[3] Siqi Nie, Denis D. Mauá, Cassio P. de Campos, Qiang Ji. Advances in learning bayesian networks of bounded treewidth. *CoRR*, abs/1406.1411, 2014.

Primeiramente, o que são *k-trees*?

Uma *k*-**tree** é definida da seguinte forma recursiva[4]:

- Um grafo completo com k vértices é uma *k-tree*.
- Se $T'_k = (V, E)$ é uma *k-tree*, $K \subseteq V$ é um k-clique e $v \notin V$, então $T_k = (V \cup \{v\}, E \cup \{(v, x) \mid x \in K\})$ é uma *k-tree*.

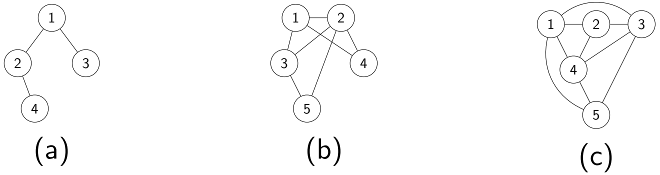

Figura: **(a)** Uma 1-*tree* (ou seja, uma árvore comum) com 4 vértices. **(b)** Uma 2-*tree* com 5 vértices. **(c)** Uma 3-*tree* com 5 vértices.

[4] Frank Harary, Edgar M. Palmer. On acyclic simplicial complexes. *Mathematika*, 15:115–122, 1968.

k-trees enraizadas

Uma *k*-**tree enraizada** é uma *k-tree* com um *k*-clique destacado $R = \{r_1, r_2, \cdots, r_k\}$ que é chamado de **raiz** da *k-tree* enraizada.

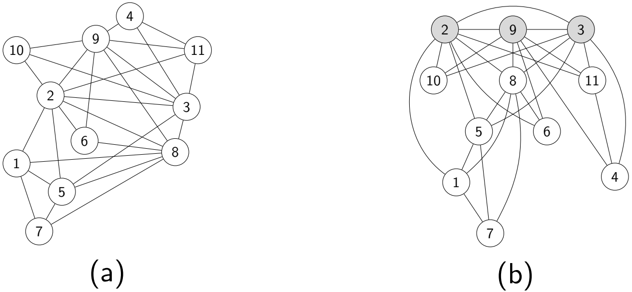

Figura: **(a)** Uma 3-*tree* T_3 com 11 vértices. **(b)** A mesma 3-*tree* (T_3) enraizada no 3-clique $\{2, 3, 9\}$.

A relação entre geração e codificação

O problema de gerar *k-trees* está intimamente relacionado ao problema de codificá-las e decodificá-las. De fato, se há uma codificação bijetiva que associa *k-trees* a *strings*, basta gerar *strings* uniformemente aleatórias para gerar *k-trees* uniformemente aleatórias.

Codificação de *k-trees*

- Em 1889, Cayley[5] demonstrou que para um conjunto de n vértices existem n^{n-2} árvores possíveis. Desde lá, foram criados vários códigos para árvores, como o de Prüfer[6].
- Em 1970, Rényi e Renýi apresentaram uma codificação redundante (ou seja, não bijetiva) para um subconjunto de *k-trees* rotuladas que chamamos de *k-trees* de Rényi[7]. Definição: Uma ***k*-tree de Rényi** R_k é uma *k-tree* enraizada com n vértices rotulados em $[1, n]$ e raiz $\{n-k+1, \cdots, n\}$.

[5] Arthur Cayley. A theorem on trees. *Quart J. Math*, 23:376–378, 1889.

[6] Heinz Prüfer. Neuer beweis eines satzes über permutationen. *Archiv der Mat. und Physik*, 27:142–144, 1918.

[7] C. Rényi, A. Rényi. The prüfer code for k-trees. *Combinatorial Theory and its Applications*, 945–971, 1970.

A solução de Caminiti *et al.*

- Apenas em 2008 surgiu um código bijetivo para *k-trees* com algoritmos lineares de codificação e decodificação. Esses algoritmos, propostos por Caminiti *et al.*, foram implementados neste trabalho.
- O código é formado por uma permutação de tamanho k e uma generalização do *Dandelion Code*[8]. A codificação das *k-trees* associa elementos em $\mathcal{T}_k^n$ (conjunto das *k-trees* com n vértices) com elementos em:

$$\mathcal{A}_k^n = \binom{[1,n]}{k} \times (\{(0,\varepsilon)\} \cup ([1,n-k] \times [1,k]))^{n-k-2}$$

[8] Ömer Eğecioğlu, J. B. Remmel. Bijections for cayley trees, spanning trees, and their q-analogues. *Journal of Combinatorial Theory*, 42:15–30, 1986.

A solução de Caminiti *et al.*

Geração uniforme de *k-trees*

Testes

Redes bayesianas

Aprendizado de redes bayesianas

Aprendizado por amostragem de *k-trees*

Experimentos

Conclusão

Agradecimentos

Perguntas?