

Obtendo Concorrência Mínima através de Ciclos Maximais sob a Dinâmica de Escalonamento por Reversão de Arestas

Carlos Eduardo Marciano

Orientadores: Felipe M. G. França & Luidi G. Simonetti

Universidade Federal do Rio de Janeiro

Defesa de Monografia

`cemarciano@poli.ufrj.br`

Web: `carloseduardov8.github.io`

14 de março de 2019

Roteiro

- 1 Introdução
- 2 SER
- 3 Concorrência Mínima
- 4 Aplicação Musical
- 5 Conclusão

Motivação

- O *Jantar dos Filósofos*: proposto por *Edsger Dijkstra* em 1965 para ilustrar *deadlocks*, *starvation* e condições de corrida.
- Variante com dois estados possíveis: “comendo” (consumindo recursos) ou “com fome” (pronto para comer).

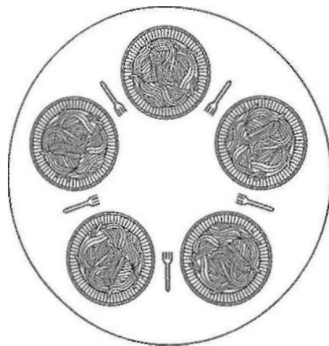


Figura 1:
O Jantar dos Filósofos [1].

Grafo de Recursos

- Nós codificam **processos** a serem escalonados.
- Arestas representam **recursos compartilhados** entre dois nós.
- Como escalonar nós a fim de **garantir justiça e prevenir problemas clássicos** de escalonamento?

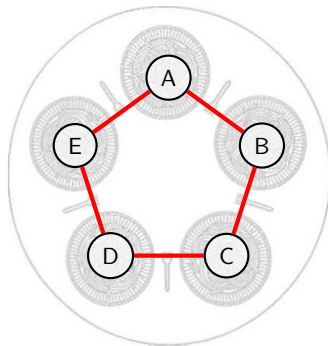


Figura 2: Grafo de recursos para o *Jantar dos Filósofos*.

Escalonamento por Reversão de Arestas (SER) [2]

- Solução distribuída para sistemas de alta carga restringidos pela vizinhança.
- Orientação acíclica: *sumidouros* operam ao mesmo tempo e reverterem suas arestas, formando novos *sumidouros*.
- Justiça: nós operam um mesmo número de vezes dentro de um período.

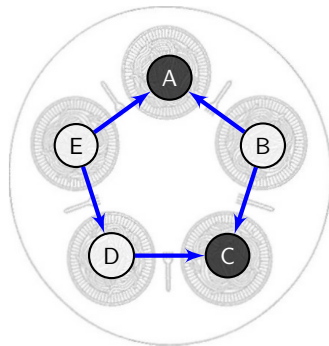


Figura 3: DAG representando o Jantar dos Filósofos.

Exemplo SER

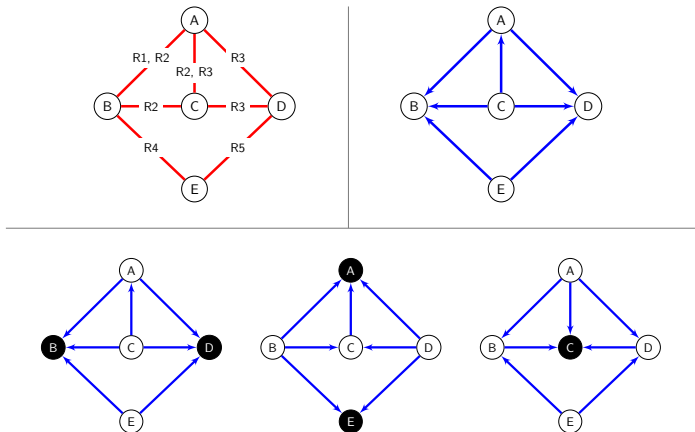
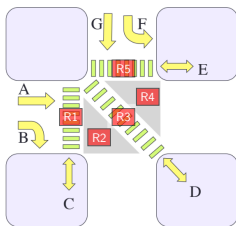
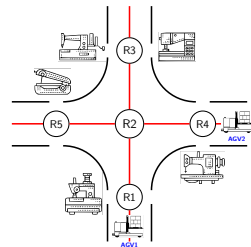
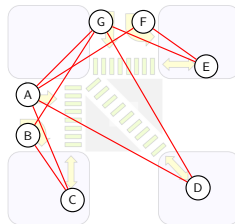


Figura 4: Grafo de recursos orientado (*sup.*) e período induzido pelo algoritmo (*inf.*).

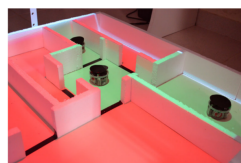
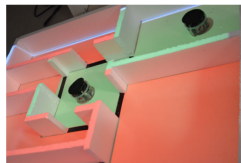
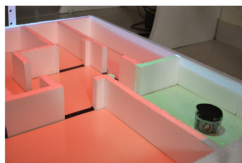
Aplicações



(a) Cruzamentos [3].



(b) Roteamento de AGVs [4].



(c) Combate ao incêndio por robôs autônomos [5].

Figura 5: Aplicações com *SER*.

Definições

Definição: Ciclo Simples

Para $G = (V, E)$, um ciclo simples $\kappa \subseteq V$ é um conjunto de vértices que formam a sequência $i_0, i_1, \dots, i_{|\kappa|-1}, i_0$. Define-se K como o conjunto de todos os ciclos simples de G .

Definições

Definição: Ciclo Simples

Para $G = (V, E)$, um ciclo simples $\kappa \subseteq V$ é um conjunto de vértices que formam a sequência $i_0, i_1, \dots, i_{|\kappa|-1}, i_0$. Define-se K como o conjunto de todos os ciclos simples de G .

Definição: Orientação Acíclica

Uma orientação acíclica de G é uma função $\omega : E \rightarrow V$ tal que nenhum ciclo κ da forma $i_0, i_1, \dots, i_{|\kappa|-1}, i_0$ existe para o qual $\omega(i_0, i_1) = i_1, \omega(i_1, i_2) = i_2, \dots, \omega(i_{|\kappa|-1}, i_0) = i_0$. Seja Ω o conjunto de todas as orientações acíclicas de G .

Definições

Definição: Ciclo Simples

Para $G = (V, E)$, um ciclo simples $\kappa \subseteq V$ é um conjunto de vértices que formam a sequência $i_0, i_1, \dots, i_{|\kappa|-1}, i_0$. Define-se K como o conjunto de todos os ciclos simples de G .

Definição: Orientação Acíclica

Uma orientação acíclica de G é uma função $\omega : E \rightarrow V$ tal que nenhum ciclo κ da forma $i_0, i_1, \dots, i_{|\kappa|-1}, i_0$ existe para o qual $\omega(i_0, i_1) = i_1, \omega(i_1, i_2) = i_2, \dots, \omega(i_{|\kappa|-1}, i_0) = i_0$. Seja Ω o conjunto de todas as orientações acíclicas de G .

Definição: Sentido de Orientação

Definimos como $n_{cw}(\kappa, \omega)$ o número de arestas no ciclo κ orientadas por ω no sentido horário, e $n_{ccw}(\kappa, \omega)$ como as orientadas no sentido anti-horário.

Concorrência

Definição: Concorrência (1)

Seja m o número de vezes que cada nó opera em um período do algoritmo *SER*. Seja p o comprimento de um período, medido em orientações. Para $G = (V, E)$, definimos *concorrência* como uma função $\gamma : \Omega \rightarrow \mathbb{R}$ tal que:

$$\gamma(\omega) = \frac{m}{p} \quad (1)$$

Concorrência

Definição: Concorrência (1)

Seja m o número de vezes que cada nó opera em um período do algoritmo *SER*.
Seja p o comprimento de um período, medido em orientações. Para $G = (V, E)$, definimos *concorrência* como uma função $\gamma : \Omega \rightarrow \mathbb{R}$ tal que:

$$\gamma(\omega) = \frac{m}{p} \quad (1)$$

Definição: Concorrência (2)

Alternativamente, para $G = (V, E)$, definimos *concorrência* como:

$$\gamma(\omega) = \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \quad (2)$$

Exemplo SER (reprise)

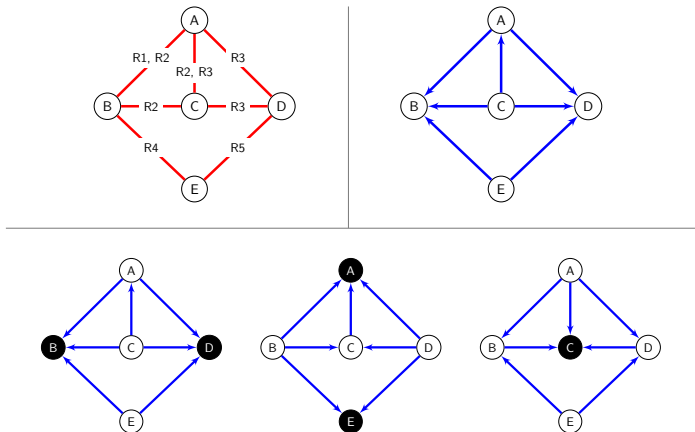


Figura 6: Concorrência: $\gamma(\omega) = m/p$; ou $\gamma(\omega) = \min_{\kappa \in K} \left\{ \frac{\min\{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\}$.

Roteiro

1 Introdução

2 SER

3 Concorrência Mínima

4 Aplicação Musical

5 Conclusão

Concorrência Mínima via Ciclos Maximais (1)

- NP-Completo [6]: Minimizar $\gamma(\omega)$ sobre todo o conjunto Ω :

$$\gamma^* = \min_{\omega \in \Omega} \left\{ \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \right\} \quad (3)$$

Concorrência Mínima via Ciclos Maximais (1)

- NP-Completo [6]: Minimizar $\gamma(\omega)$ sobre todo o conjunto Ω :

$$\gamma^* = \min_{\omega \in \Omega} \left\{ \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \right\} \quad (3)$$

Lema 1

$$\gamma^* = \min_{\kappa \in K} \left\{ \frac{1}{|\kappa|} \right\}$$

Concorrência Mínima via Ciclos Maximais (1)

- NP-Completo [6]: Minimizar $\gamma(\omega)$ sobre todo o conjunto Ω :

$$\gamma^* = \min_{\omega \in \Omega} \left\{ \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \right\} \quad (3)$$

Lema 1

$$\gamma^* = \min_{\kappa \in K} \left\{ \frac{1}{|\kappa|} \right\}$$

Demonstração

Relembre a definição de concorrência: $\gamma(\omega) = \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\}$.

Para um dado ω' , seja κ' o ciclo escolhido pelo minimizador da definição de concorrência. Seja $x = \min \{n_{cw}(\kappa', \omega'), n_{ccw}(\kappa', \omega')\}$. Logo, temos $\gamma(\omega) = x/|\kappa'|$.

1/3

Concorrência Mínima via Ciclos Maximais (1)

- NP-Completo [6]: Minimizar $\gamma(\omega)$ sobre todo o conjunto Ω :

$$\gamma^* = \min_{\omega \in \Omega} \left\{ \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \right\} \quad (3)$$

Lema 1

$$\gamma^* = \min_{\kappa \in K} \left\{ \frac{1}{|\kappa|} \right\}$$

Demonstração

Porém, para qualquer ciclo $\kappa \in K$, é possível orientar κ com algum $\omega \in \Omega$ de forma que $n_{cw}(\kappa, \omega) = 1$ e $n_{ccw}(\kappa, \omega) = |\kappa| - 1$, ou vice-versa. Logo, se ω' , aplicado a κ' , não produziu o valor $x = 1$, haverá outra orientação $\omega \in \Omega$ que produzirá $\gamma(\omega) = 1/|\kappa'|$.

2/3

Concorrência Mínima via Ciclos Maximais (1)

- NP-Completo [6]: Minimizar $\gamma(\omega)$ sobre todo o conjunto Ω :

$$\gamma^* = \min_{\omega \in \Omega} \left\{ \min_{\kappa \in K} \left\{ \frac{\min \{n_{cw}(\kappa, \omega), n_{ccw}(\kappa, \omega)\}}{|\kappa|} \right\} \right\} \quad (3)$$

Lema 1

$$\gamma^* = \min_{\kappa \in K} \left\{ \frac{1}{|\kappa|} \right\}$$

Demonstração

Suponha que γ^* , a concorrência mínima de G , seja menor que $1/|\kappa'|$. Se isto for verdade, deverá existir um ciclo κ^* que, sob alguma orientação ω^* , produzirá $1/|\kappa^*| < 1/|\kappa'|$. Logo, encontrar γ^* tornou-se um problema de minimização sobre todo $\kappa \in K$. □

Concorrência Mínima via Ciclos Maximais (2)

- Resta encontrar ω^* tal que $\gamma^* = \gamma(\omega^*)$.

Concorrência Mínima via Ciclos Maximais (2)

- Resta encontrar ω^* tal que $\gamma^* = \gamma(\omega^*)$.

Teorema 1

Dado qualquer ciclo maximal $\kappa^* \in K$ como entrada, existe um algoritmo de complexidade linear para encontrar uma orientação $\omega^* \in \Omega$ tal que $\gamma(\omega^*)$ é mínimo para todo $\omega \in \Omega$.

Concorrência Mínima via Ciclos Maximais (2)

- Resta encontrar ω^* tal que $\gamma^* = \gamma(\omega^*)$.

Teorema 1

Dado qualquer ciclo maximal $\kappa^* \in K$ como entrada, existe um algoritmo de complexidade linear para encontrar uma orientação $\omega^* \in \Omega$ tal que $\gamma(\omega^*)$ é mínimo para todo $\omega \in \Omega$.

Demonstração

Pela prova do Lemma 1, para atingir γ^* , deve-se orientar κ^* tal que $n_{cw}(\kappa^*, \omega^*) = 1$ e $n_{ccw}(\kappa^*, \omega^*) = |\kappa^*| - 1$ (ou vice-versa). Isto pode ser realizado em tempo linear ao percorrermos o ciclo κ^* e atribuímos um número de identificação crescente $1, \dots, |\kappa^*|$ para cada vértice visitado, resultando em uma ordenação topológica do ciclo. Por fim, orienta-se as arestas no sentido dos vértices de maior identificador, cumprindo o requisito.

1/2

Concorrência Mínima via Ciclos Maximais (2)

- Resta encontrar ω^* tal que $\gamma^* = \gamma(\omega^*)$.

Teorema 1

Dado qualquer ciclo maximal $\kappa^* \in K$ como entrada, existe um algoritmo de complexidade linear para encontrar uma orientação $\omega^* \in \Omega$ tal que $\gamma(\omega^*)$ é mínimo para todo $\omega \in \Omega$.

Demonstração

Resta orientar os demais vértices de G tal que ω^* sempre será de fato acíclica. Seja $S = V - \kappa^*$ o conjunto dos vértices restantes de G . Atribui-se um número de identificação crescente $|\kappa^*| + 1, \dots, |V|$ para cada vértice em S , e então orienta-se todas as arestas de G na direção dos vértices com maior identificador. Por absurdo, se ω^* possuir ciclos, existirá um caminho direcionado i_0, i_1, \dots, i_0 . No entanto, como $id[i_0] > id[i_1]$, é impossível retornar a i_0 após a partida, para qualquer $i_0 \in V$. Portanto, nenhum ciclo será formado. \square

Viabilidade Computacional

- Implementação do **modelo para o *Simple Cycle Problem*** [7]:

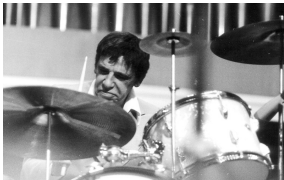
Nós	Arestas	p	$ \kappa^* $	Conc. Mín.	Tempo CPU (s)
200	392	0.01	183	1/183	1
200	3826	0.1	200	1/200	2
1000	1912	0.002	882	1/882	169
1000	19912	0.02	1000	1/1000	552
1000	180151	0.2	-	-	> 3600
2000	4079	0.001	1807	1/1807	874
2000	40034	0.01	2000	1/2000	3599
2000	380147	0.1	-	-	> 3600
2000	1999000	1	-	-	> 3600

Tabela 1: Experimentos para encontrar a concorrência mínima de grafos circulantes gerados aleatoriamente.

Roteiro

- 1 Introdução
- 2 SER
- 3 Concorrência Mínima
- 4 Aplicação Musical**
- 5 Conclusão

Contexto Musical



(a) Buddy Rich, jazz.



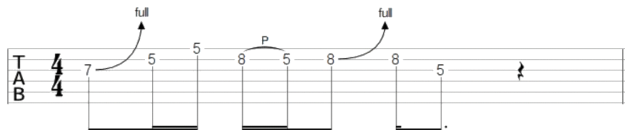
(b) Joe Bonamassa, blues.

Figura 7: Virtuosos (*Creative Commons*).

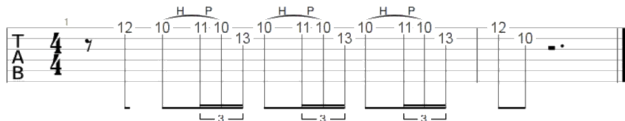
- A **geração de melodias por computador** tem sido estudada desde a década de 50 [8].
- Duas abordagens: explícita (em que as **regras de composição são especificadas por humanos**) e implícita [9].
- Música ocidental: tem como característica o *contraponto* (ou *polifonia*), com **múltiplas vozes melódicas** [10].

Frases Musicais

- Em *blues*, *jazz* e *rock*, é comum existir uma **dinâmica de “pergunta e resposta”** com frases musicais.



(a) Frase antecedente.



(b) Frase consequente.

Figura 8: Tablaturas de frases musicais [11].

Montando Faixas de Máxima Duração

- Gostaríamos que nosso modelo capturasse as seguintes restrições:
 - Uma frase *consequente* **apenas pode ser tocada** após uma *antecedente*, formando um *lick*;
 - Apenas frases do mesmo tipo (*antecedente* ou *consequente*) podem **tocar ao mesmo tempo**;
 - Frases de **diferentes intensidades** (e.g. número de notas) podem não soar bem juntas;
 - A composição final deve ser um *loop*, incluir todas as frases e ser de **máxima duração**.

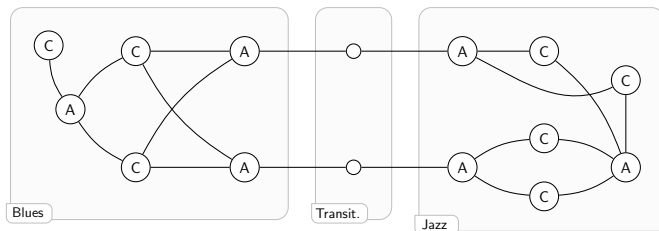
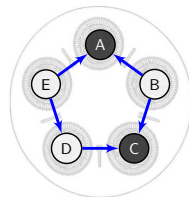


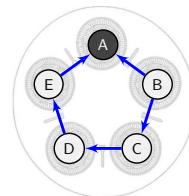
Figura 9: Exemplo de modelagem.

Conclusão

- Contribuições: estratégia computacional para a **obtenção de concorrência mínima** e nova proposta para a **criação de faixas musicais**.
- Padrão *MIDI*: **faixas com duração de horas** e potencial fonte de inspiração para artistas.
- Trabalhos futuros: elaboração de um modelo computacional para a **obtenção de concorrência máxima** sob *SER*.



(a) Concorrência máxima.



(b) Concorrência mínima.

Figura 10: Concorrências extremas.

Agradecimentos

Obrigado!

Perguntas & Respostas

Bibliografia I

- [1] TANENBAUM, A. S., *Modern Operating Systems*.
3rd ed., pp. 143–165.
Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2007.
- [2] BARBOSA, V. C., GAFNI, E., “Concurrency in heavily loaded neighborhood-constrained systems”, *ACM Trans. on Program. Lang. and Syst.*, v. 11, no. 4, pp. 562–584, 1989.
- [3] CARVALHO, D., PROTTI, F., DE GREGORIO, M., et al., “A Novel Distributed Scheduling Algorithm for Resource Sharing Under Near-Heavy Load”, *Lecture Notes in Computer Science*, v. 3544, pp. 431–442, 2004.
- [4] Lengerke, O., ACUÑA, H. G., DUTRA, M. S., et al., “Distributed control of job-shop systems via edge reversal dynamics for automated guided vehicles”, *1st International Conference on Intelligent Systems and Applications*, pp. 25–30, 2012.
- [5] ALVES, D. S. F., SOARES, E. E., STRACHAN, G. C., et al., *A Swarm Robotics Approach to Decontamination. In: Mobile Ad Hoc Robots and Wireless Robotic Systems: Design and Implementation*.
1st ed., pp. 107–122.
Hershey, PA, USA: IGI Publishing Hershey, 2012.
- [6] ARANTES JR, G. M., *Trilhas, Otimização de Concorrência e Inicialização Probabilística em Sistemas sob Reversão de Arestas*, Ph.D. Thesis, Prog. de Eng. de Sist. e Comp., Univ. Fed. do Rio de Janeiro, 2006.

Bibliografia II

- [7] LUCENA, A., DA CUNHA, A. S., SIMONETTI, L., "A New Formulation and Computational Results for the Simple Cycle Problem", *Electronic Notes in Discrete Mathematics*, v. 44, no. 5, pp. 83–88, 2013.
- [8] NIERHAUS, G., *Algorithmic Composition: Paradigms of Automated Music Generation*. Springer-Verlag: Vienna, Austria, 2009.
- [9] SHAN, M.-K., CHIU, S.-C., "Algorithmic compositions based on discovered musical patterns", *Multimedia Tools and Applications*, v. 46, n. 1, pp. 1–23, Jan. 2010.
- [10] SCHMIDT-JONES, C., *Understanding Basic Music Theory*. OpenStax CNX: Houston, TX, USA, 2007.
- [11] BELL, J., *144 Blues Guitar Licks*. JamString: East Midlands, UK, 2015, mobile application. Version 15.41942290.