Técnicas de Controle de Concorrência

André Luis Schwerz
andreluis@utfpr.edu.br

Universidade Tecnológica Federal do Paraná

Banco de Dados 2
2020/1

Agenda

1. Como encontrar schedules serializáveis
2. Técnicas Pessimistas
3. Técnicas Otimistas

Objetivos

Entender:

- Como encontrar *schedules* serializáveis
- O que são técnicas otimistas
- O que são técnicas pessimista

Agenda

1. Como encontrar schedules serializáveis
2. Técnicas Pessimistas
3. Técnicas Otimistas

- **Pessimistas:**
  - Supõem que sempre ocorre interferência entre transações
  - A serializabilidade é garantida enquanto a transação está ativa
  - Técnicas:
    - **Bloqueio** (locking)
    - Timestamp
    - Multi-versão
- **Otimistas:**
  - Supõem que quase nunca ocorre interferência entre transações
  - A serializabilidade é verificada somente ao final de uma transação
  - Técnica:
    - **Validação**

Agenda

Técnicas Pessimistas

Técnicas baseadas em bloqueio

- Técnicas mais utilizadas pelos SGBDs
- Princípio de funcionamento
  - Controle de operações read(X) e write(X) e postergação (através de bloqueio) de algumas dessas operações de modo a evitar conflito.
  - Um bloqueio é uma variável associada a um item de dado que descreve seu status no sistema.
- Todo dado possui um **status de bloqueio**
  - **Binário - Ruim**
    - **Liberado** (Unlocked– U)
    - **Bloqueado** (Locked – L)
  - **Compartilhados/Exclusivos - Boa**
    - **Liberado** (Unlocked– U)
    - Com **bloqueio compartilhado** (Shared lock - S)
    - Com **bloqueio exclusivo** (Exclusive lock - X)

Técnicas Pessimistas

Tipos de Bloqueio

- **Bloqueio Compartilhado (S)**
  - Solicitado por uma transação que deseja realizar leitura de um dado D
    - Várias transações podem manter esse bloqueio sobre D
- **Bloqueio Exclusivo (X)**
  - Solicitado por uma transação que deseja realizar leitura + atualização de um dado D
    - Uma única transação pode manter esse bloqueio sobre D
- Matriz de Compatibilidade de Bloqueios

	S	**X**
S	verdadeiro	falso
X	falso	falso

- Informações de bloqueio são mantidas no DD

`<ID-dado,status-bloqueio,ID-transação>`

Técnicas Pessimistas

Operações de bloqueio na *schedule*

- O *scheduler* gerencia bloqueios por meio da invocação automática de operações de bloqueio, conforme a operação que a transação deseja realizar em um dado.
- Operações:
  - **ls(D)** $\rightarrow$ solicitação de bloqueio compartilhado sobre D
  - **lx(D)** $\rightarrow$ solicitação de bloqueio exclusivo sobre D
  - **u(D)** $\rightarrow$ libera o bloqueio sobre D

Técnicas Pessimistas

Exemplo de um schedule com bloqueios

$H = ls_1(X); r_1(X); u_1(X); lx_2(X); w_2(X); u_2(X); c_2; lx_1(X); w_1(X); u_1(X); c_1;$

T_1	T_2
lock-S(X)	
read(X)	
unlock(X)	
	lock-X(X)
	write(X)
	unlock(X)
	commit()
lock-X(X)	
write(X)	
unlock(X)	
commit( )	

- Solicitação de bloqueio compartilhado:

```
lock-S(D, Tx)
    se lock(D) = 'U' então
        insere Tx na lista-READ(D);
        lock(D) = 'S'
    senão se lock(D) = 'S' então
        insere Tx na lista-READ(D);
    senão /* lock(D) = 'X' */
        insere (Tx, 'S') no fila-WAIT(D);
    fim
fim
```

  - lock(D): status de bloqueio de D
  - lista – READ(D): transações com bloqueio compartilhado sobre D
  - fila – WAIT(D): transações aguardando a liberação de um bloqueio conflitante sobre D

- Solicitação de bloqueio exclusivo:

```
lock-X(D, Tx)
    se lock(D) = 'U' então
        lock(D) = 'X'
    senão
        insere (Tx, 'X') no fila-WAIT(D);
    fim
fim
```

  - lock(D): status de bloqueio de D
  - fila − WAIT(D): transações aguardando a liberação de um bloqueio conflitante sobre D

- Solicitação de desbloqueio:

```
unlock(D, Tx)
    se lock(D) = 'X' então
        lock(D) = 'U'
        desperta fila-WAIT(D);
    senão se lock(D) = 'S' então
        remove Tx na lista-READ(D);
        se lista-READ(D) vazia então
            lock(D) = 'U'
            desperta fila-WAIT(D);
        fim
    fim
fim
```

  - lock(D): status de bloqueio de D
  - lista – READ(D): transações com bloqueio compartilhado sobre D
  - fila – WAIT(D): transações aguardando a liberação de um bloqueio conflitante sobre D

Técnicas Pessimistas

Uso de bloqueios "S" e "X"

- Não garantem escalonamentos serializáveis
- Exemplo:

$$H = ls_1(X); r_1(X); u_1(X); lx_2(X); w_2(X); u_2(X); c_2; lx_1(X); w_1(X); u_1(X); c_1;$$

- Necessita-se de uma técnica mais rigorosa de bloqueio para garantir a serializabilidade
  - Técnica mais utilizada:
    - **Bloqueio bifásico** (*Two-Phase Locking* – **2PL**)

Técnicas Pessimistas

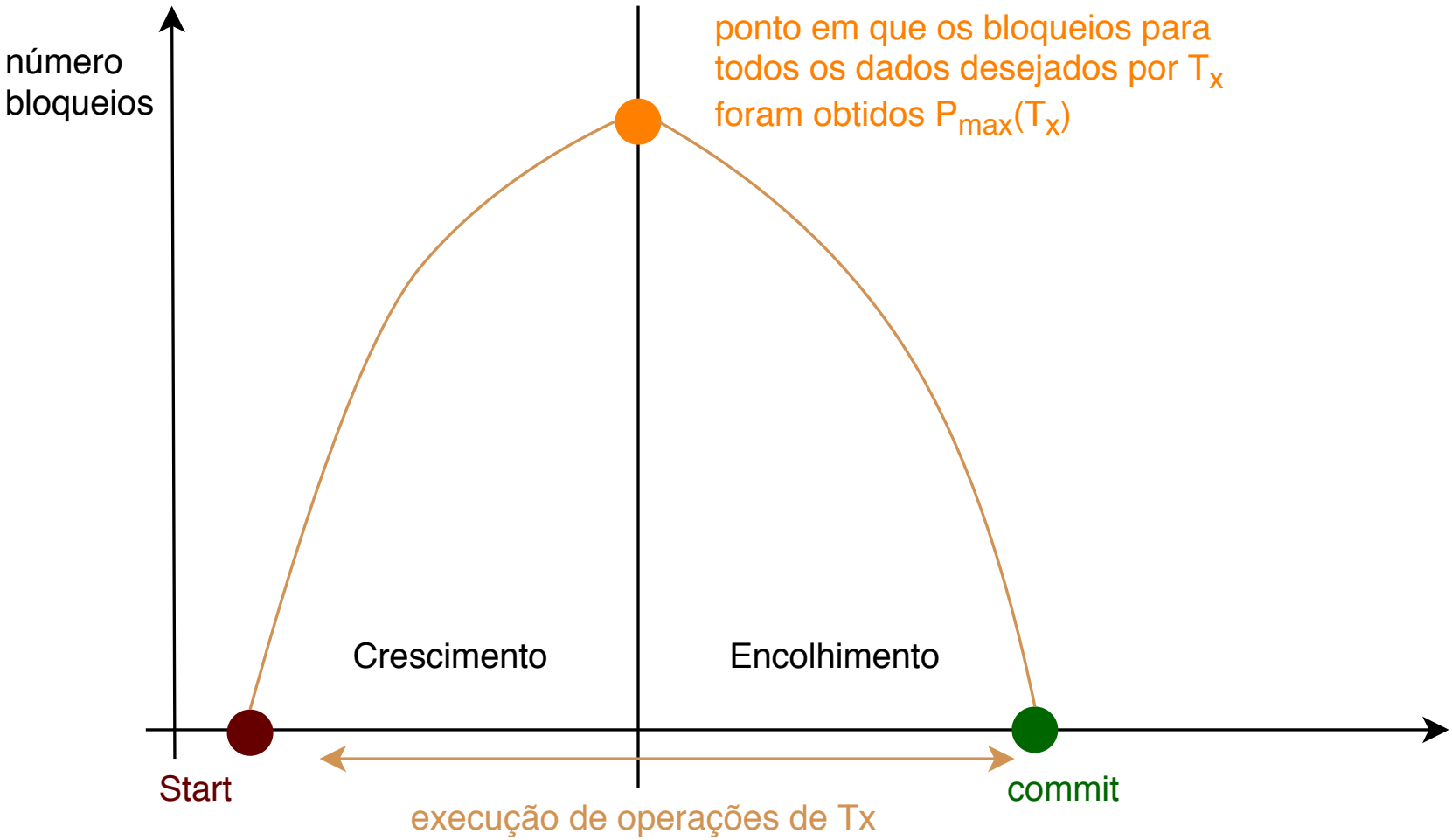
Protocolo de Bloqueio Bifásico (2PL)

- Premissa:
  - "para toda transação T_x, todas as operações de bloqueio de dados feitas por T_x precedem a primeira operação de desbloqueio feita por T_x"
- Protocolo de duas fases:
  - Fase de expansão ou crescimento
    - T_x pode obter bloqueios, mas não pode liberar nenhum bloqueio
  - Fase de retrocesso ou encolhimento
    - T_x pode liberar bloqueios, mas não pode obter nenhum bloqueio

Técnicas Pessimistas

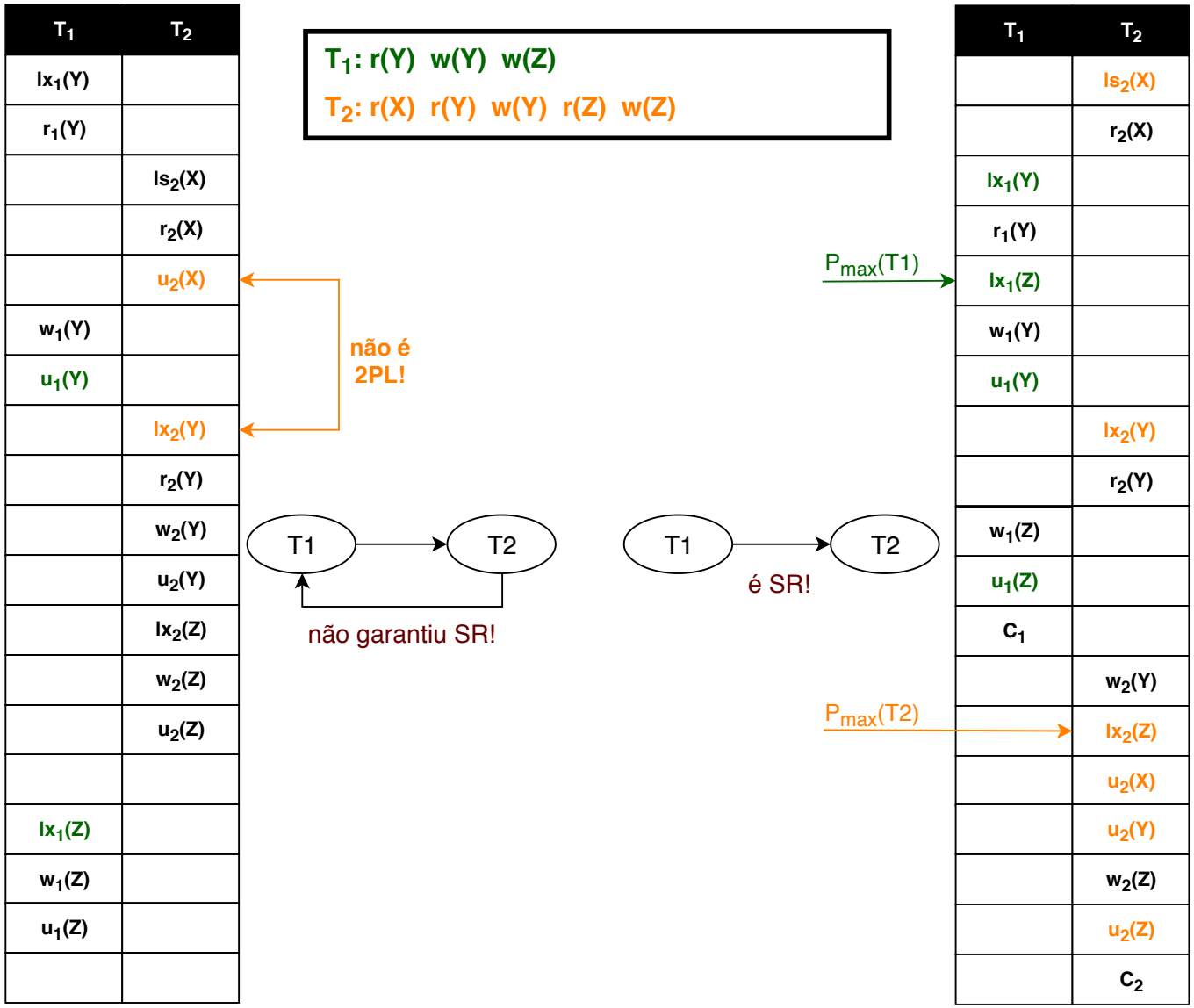
Protocolo de Bloqueio Bifásico (2PL)

Gráfico de bloqueios de Tx



Técnicas Pessimistas

Protocolo de Bloqueio Bifásico (2PL)



- Técnica que sempre garante escalonamentos SR sem a necessidade de se construir um grafo de dependência para teste.
  - Se T_x alcança P_{max}, T_x não sofre interferência de outra transação T_y, pois se T_y deseja um dado de T_x em uma operação que poderia gerar conflito com T_x, T_y tem que esperar (evita ciclo $T_y \rightarrow T_x$)
  - Depois que T_x liberar os seus dados, não precisará mais deles, ou seja, T_x não interferirá nas operações feitas futuramente nestes dados por T_y (evita também ciclo $T_y \rightarrow T_x$)

- Limita a concorrência
  - Se um dado pode permanecer bloqueado por T_x muito tempo até que T_x adquira bloqueios em todos os outros dados que deseja
- Não garante escalonamentos
  - Livres de *deadlock*
    - T_x espera pela liberação de um dado bloqueado por T_y de forma conflitante e vice-versa
  - Adequados à recuperação
    - Há dependência
    - Portanto, há aborto em cascata.

Técnicas Pessimistas

Exercício

1. Apresente um início de escalonamento 2PL que recaia em uma situação de *deadlock*
2. Apresente um escalonamento 2PL que não seja recuperável

Lembrete

Um escalonamento é recuperável se T_x nunca executa *commit* antes de T_y, caso T_x tenha lido dados atualizados por T_y

Técnicas Pessimistas

2PL Estrito

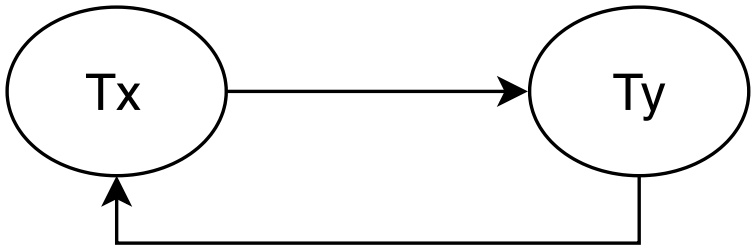
- A transação T_x não liberará nenhum de seus bloqueios exclusivos antes do seu ponto de confirmação.
- Logo, nenhuma outra transação pode ler ou gravar um item de dado escrito por T_x a menos que T_x tenha confirmado.
- **Vantagem**
  - Obtém *schedules* que são estritos
  - Mais facilmente recuperáveis (veja aula passada)
- **Desvantagem**
  - Não evita *deadlock*.

Técnicas Pessimistas

2PL Rigoroso

- A transação T_x não liberará nenhum de seus bloqueios exclusivos ou compartilhados antes do seu ponto de confirmação.
- Logo, nenhuma outra transação pode ler ou gravar um item de dado lido ou escrito por T_x a menos que T_x tenha confirmado.
- **Vantagem**
  - Obtém schedules que são estritos, entretanto, mais rigoroso.
  - Mais facilmente recuperáveis (veja aula passada)
- **Desvantagem**
  - Não evita *deadlock*.

- Ocorrência de *deadlock*
  - T_y está na `fila` – `WAIT(D1)` de um dado `D1` bloqueado por T_x
  - T_x está na `fila` – `WAIT(D2)` de um dado `D2` bloqueado por T_y
- Pode ser descoberto através de um grafo de espera de transações
- Se o grafo é cíclico existe *deadlock*



Técnicas Pessimistas

Lidando com o *deadlock*

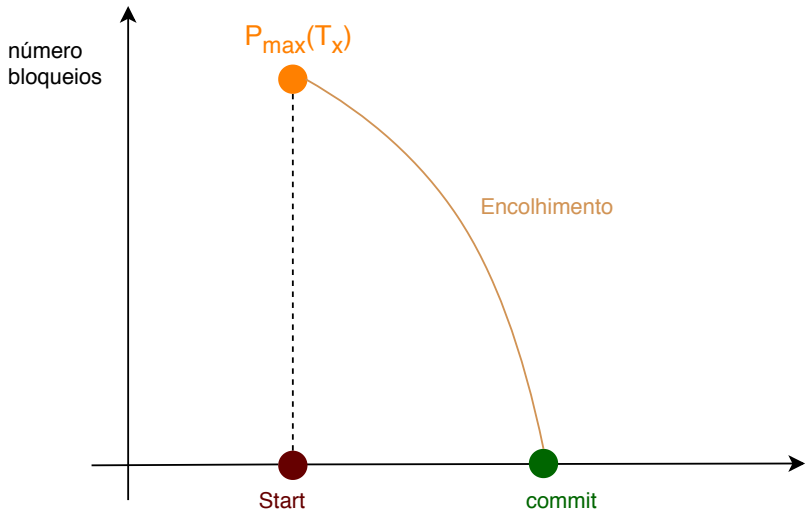
- *Deadlocks* ocorrem com frequência!
- O que fazer: **Tratamento ou Prevenção?**

1. Prevenção:
   - A prevenção impõem um *overhead* no processamento de transações porque controles adicionais devem ser usados para evitar o *deadlock*
   - Estratégias:
     - Técnica de bloqueio 2PL conservador
     - Técnica de espera-cautelosa (*cautious-wait*)
     - Técnicas baseadas em *timestamp* (*wait-die* e *wound-wait*)
2. Tratamento:
   - O tratamento também impõem algum *overhead* no processamento de transações.
   - Estratégia:
     - **Uso de *timeout***
     - se tempo de espera de $T_x >$ timeout $\Longrightarrow$ abort(T_x)

Técnicas Pessimistas

2PL Conservador

- T_x deve bloquear todos os dados que deseja antes de iniciar qualquer operação
  - caso não seja possível bloquear todos os dados, nenhum bloqueio é feito e T_x entra em espera para tentar novamente



- **Vantagem**
  - uma vez adquiridos todos os seus bloqueios, T_x não entra em *deadlock* durante a sua execução
- **Desvantagem**
  - espera pela aquisição de todos os bloqueios

Técnicas Pessimistas

Técnica *Cautious-Waiting*

- Princípio de Funcionamento

```
se Tx deseja D e D está bloqueado por Ty então
    se Ty não está em alguma fila-WAIT então
        Tx espera
    senão
        abort(Tx)
        start(Tx)
    fim
fim
```

- **Vantagem**
  - Se T_y já está em espera, T_x é abortada para evitar um possível ciclo de espera
- **Desvantagem**
  - muitos abortos podem ser provocados sem nunca ocorrer um *deadlock*
  - problema de inanição (*starvation*)

Técnicas Pessimistas

Técnicas Baseadas em *Timestamp*

- *Timestamp*:
  - Rótulo de tempo associado à T_x (TS(T_x))
- Técnicas:
  - Considere que T_x deseja um dado bloqueado por outra transação T_y.

Técnica 1: esperar-ou-morrer (wait-die)

```
se TS(Tx) < TS(Ty) então
   Tx espera
senão
    abort(Tx)
    start(Tx) com o mesmo TS
fim
```

Técnica 2: ferir-ou-esperar (wound-wait)

```
se TS(Tx) < TS(Ty) então
   abort(Ty)
   start(Ty) como o mesmo TS
senão
    Tx espera
fim
```

- **Vantagem:**
  - Evitam *starvation* (espera indefinida) de uma T_x
  - Porque quanto mais antiga for T_x, maior a sua prioridade
- **Desvantagem:**
  - Muitos abortos podem ser provocados, sem nunca ocorrer um *deadlock*

Agenda

1. Como encontrar schedules serializáveis
2. Técnicas Pessimistas
3. Técnicas Otimistas

Técnicas Otimistas

Introdução

- Técnicas pessimistas
  - *Overhead* no processamento de transações
  - Executam verificações e ações antes de qualquer operação no BD para garantir a serializabilidade (solicitação de bloqueio)
- Técnicas otimistas
  - Não realizam nenhuma verificação durante o processamento da transação
  - Pressupõem nenhuma ou pouca interferência
  - Verificações de violação de serializabilidade feitos somente ao final de cada transação
  - Técnica mais conhecida: Técnica de Validação

Técnicas Otimistas

Técnica de Validação

- Técnica na qual atualizações de uma transação T_x são feitas sobre cópias locais dos dados.
- Quando T_x solicita *commit* é feita a sua validação
- T_x violou a serializabilidade?
  - SIM: T_x é abortada e reiniciada posteriormente
  - NÃO: Atualiza o BD a partir das cópias dos dados e encerra T_x

Técnicas Otimistas

Técnica de Validação

Cada transação T_x passa por 3 etapas:

1. Leitura
   - T_x lê dados de transações confirmadas do BD e atualiza dados em cópias locais.
2. Validação
   - Análise da manutenção da serializabilidade de conflito caso as atualizações de T_x sejam efetivadas no BD.
3. Escrita
   - Se e etapa de validação for OK, aplica-se as atualizações de T_x no BD e T_x encerra com sucesso, caso contrário, T_x é abortada.

Dúvidas