Mecanismos para a manutenção do Isolamento

- Implementado pelo componente **"scheduler"** do SGBD, que:
- pode permitir a execução imediata da operação (r, w, start, commit ou abort) ou
- atrasar a execução da operação ou
- rejeitar a operação e assim abortar a transação.

Tipos de Scheduler - Funcionamento

- Agressivos: execução imediata das operações (sujeito a aborts)
- Conservadores: analisam as operações antes de liberar a execução

Protocolos

- **Protocolo baseado em bloqueios (locks)**
  - Princípio básico: ordenar conflitos postergando (bloqueando) a execução de certas operações.
  - Enquanto Ti faz acesso a um item de dado, nenhuma outra transação pode modificar aquele item de dado.

Protocolos

- **Protocolo baseado em bloqueios (locks)**
  - Modos de Bloqueio:
  - PARTILHADO (ls): Se Ti obteve um bloqueio no modo partilhado (S) no item a, então Ti pode ler este item, mas não pode gravar a.
  - EXCLUSIVO (lx): se Ti obteve um bloqueio no modo exclusivo (X) no item a, então Ti pode ler e gravar a.
  - Os bloqueios devem ser aplicados de acordo com as operações que serão executadas nos itens de dados.

Protocolos

- **Protocolo baseado em bloqueios (locks)**
  - Compatibilidade de bloqueio

	Tipo de lock que a transação possui	S	X
Tipo de lock que a transação quer obter	S	verdadeiro	falso
	X	falso	falso

Duas ou mais transações (T1 e T2) que vão ler o dado x podem ter bloqueios compartilhados sobre ele (**ls1[x] e ls2[x]**).

Uma transação que pede um bloqueio do tipo ***s*** sobre um dado x (**ls[x]**) pode solicitar uma conversão para um bloqueio do tipo ***x*** (**lx[x]**) para poder escrever sobre o dado x.

Protocolos para Controle de Concorrência

- Baseado em bloqueios:
  - 2PL
  - Multigranularidade de Bloqueios
  - Baseado em Grafos
- Sem bloqueio:
  - Ordenação por marcador de tempo

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- Modos de Bloqueio: Partilhado (ls) e Exclusivo (lx)
  - Os bloqueios e desbloqueios são emitidos em duas fases:
    - **FASE DE CRESCIMENTO:** uma transação pode obter bloqueios, entretanto não pode liberar nenhum. Operações ls[ ] e lx[ ]
    - **FASE DE ENCOLHIMENTO:** após liberado um bloqueio, a transação não pode obter nenhum novo bloqueio. Operações us[ ] e ux[ ]

Assegura a serializabilidade sem usar grafo de serialização, entretanto não impede a existência de "deadlocks"

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL básico
- 2PL conservador
- 2PL strict

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL básico
  - Regras de funcionamento:
    - uma transação T quer executar uma operação (r[] ou w[]) sobre um dado x.
    - Se T não tem um bloqueio sobre x e pode obtê-lo (em função das outras transações), T bloqueia x e executa a operação.
    - Se x já está bloqueado por outra transação, T fica em espera.
    - Uma transação tem que desbloquear um dado para que outra possa bloqueá-lo.
    - Uma transação que já desbloqueou um dado não pode bloquear mais nem um outro.

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL básico
  - ls1[x]; r1[x]; lx1[y]; w1[y]; us1[x]; lx2[x]; w2[x]; ux1[y]; c1; lx2[y]; w2[y]; ux2[x]; ux2[y]; c2

ls(x)	
r(x)	
lx(y)	
x(y)	
u(x)	
	lx(x)
	w(x)
ux(y)	
c	
	lx(y)
	w(y)
	ux(x)
	ux(y)
	c

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL conservador
  - Regras de funcionamento:
    - a transação tem que obter todos os bloqueios antes de executar qualquer operação (ela pré-declara seu conjunto de reads e writes).
    - A liberação dos bloqueios pode ser gradual, à medida que a transação utiliza os dados.
  - Vantagens:
    - evita o surgimento de deadlocks
    - nunca aborta transações
    - se uma transação vai executar e não consegue obter os bloqueios necessários, ela é colocada em espera.

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL conservador
  - Regras de funcionamento:
    - a transação tem que obter todos os bloqueios antes de executar qualquer operação (ela pré-declara seu conjunto de reads e writes).
    - A liberação dos bloqueios pode ser gradual, à medida que a transação utiliza os dados.
  - Vantagens:
    - evita o surgimento de deadlocks
    - nunca aborta transações
    - se uma transação vai executar e não consegue obter os bloqueios necessários, ela é colocada em espera.

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL conservador
  - Regras de funcionamento:
    - a transação tem que obter todos os bloqueios antes de executar qualquer operação (ela pré-declara seu conjunto de reads e writes).
    - A liberação dos bloqueios pode ser gradual à medida que a transação utiliza os dados.
  - Vantagens:
    - evita o surgimento de deadlocks
    - se uma transação vai executar e não consegue obter os bloqueios necessários, ela é colocada em espera.

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL conservador
  - pode diminuir (afetar) o paralelismo
  - Timeout curto demais : número de aborts desnecessários cresce – diminui a eficiência da máquina
  - Timeout longo demais : resolução de deadlocks mais demorada (demora mais para detectá-los)
  - WFG : manutenção (processamento + memória) é cara

- 2PL conservador

ls(x)	
lx(y)	
r(x)	
u(x)	
x(y)	
	lx(x)
ux(y)	
c	
	lx(y)
	w(x)
	w(y)
	ux(x)
	ux(y)
	c

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

- 2PL strict
  - Só permite a liberação dos bloqueios de uma transação quando ela terminar (commit/abort).
  - A obtenção dos bloqueios pode ser gradual, à medida que a transação precisa utilizar os dados.

Este mecanismo garante execuções concorrentes serializáveis & recuperáveis & ACA & Strict

Protocolo de bloqueios em 2 fases (Two Phase Lock – 2PL)

ls(x)	
r(x)	
lx(y)	
x(y)	
c	
u(x)	
u(y)	
	lx(x)
	lx(y)
	w(x)
	w(y)
	c
	ux(x)
	ux(y)

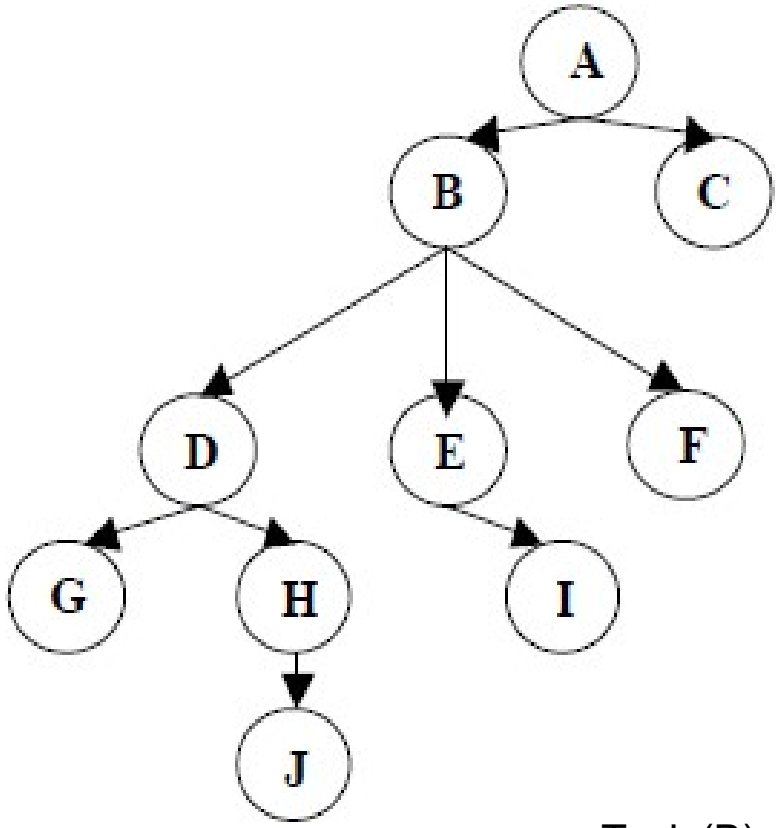
Baseado em Grafo

- Conhecimento da ordem de acesso dos itens de dados no banco de dados
  - construção de um grafo com a ordem de acesso aos itens
- Modo de bloqueio: Exclusivo (X).

Baseado em Grafo

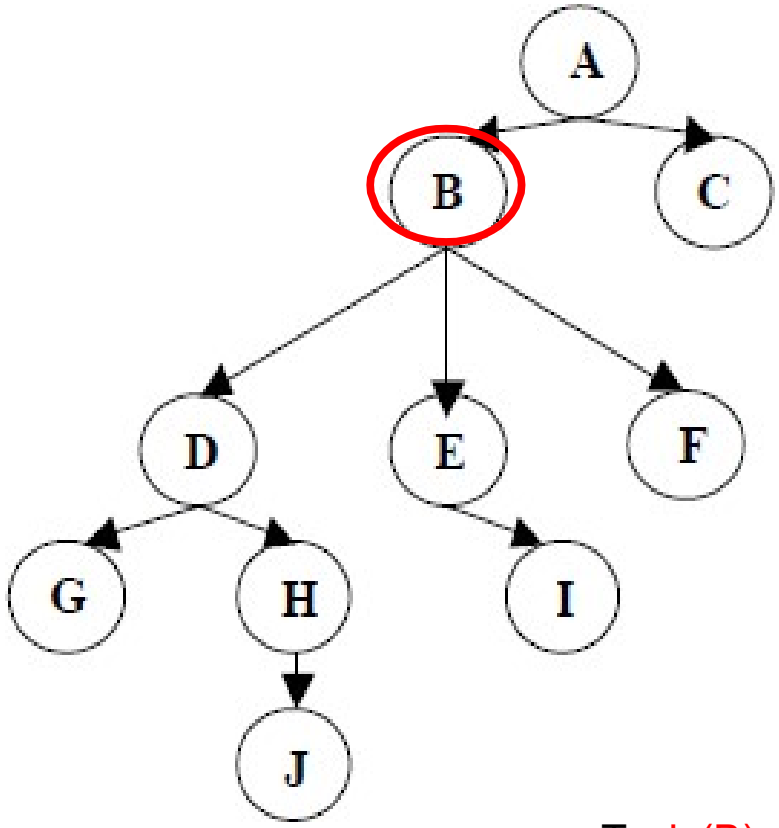
- Critérios:
  - O primeiro bloqueio de Ti pode ser em qualquer item de dado.
  - Em seguida, um item de dado A pode ser bloqueado por Ti apenas se o pai de A estiver bloqueado corretamente por Ti.
  - Itens de dados podem ser desbloqueados a qualquer instante.
  - Um item de dado que tenha sido bloqueado e desbloqueado por Ti não pode ser novamente bloqueado por Ti.

Baseado em Grafo



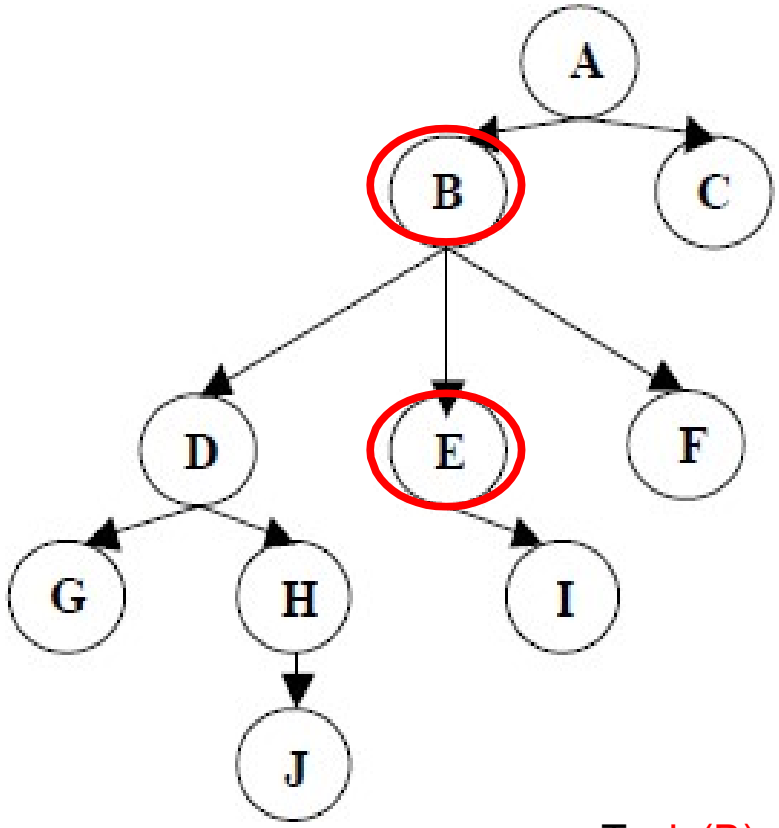
T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



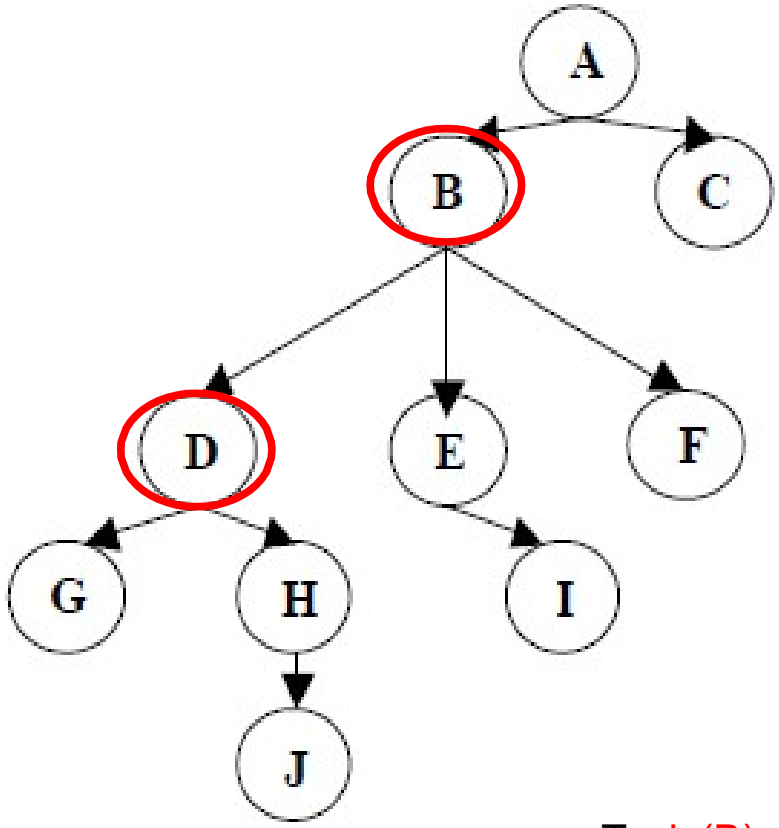
T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



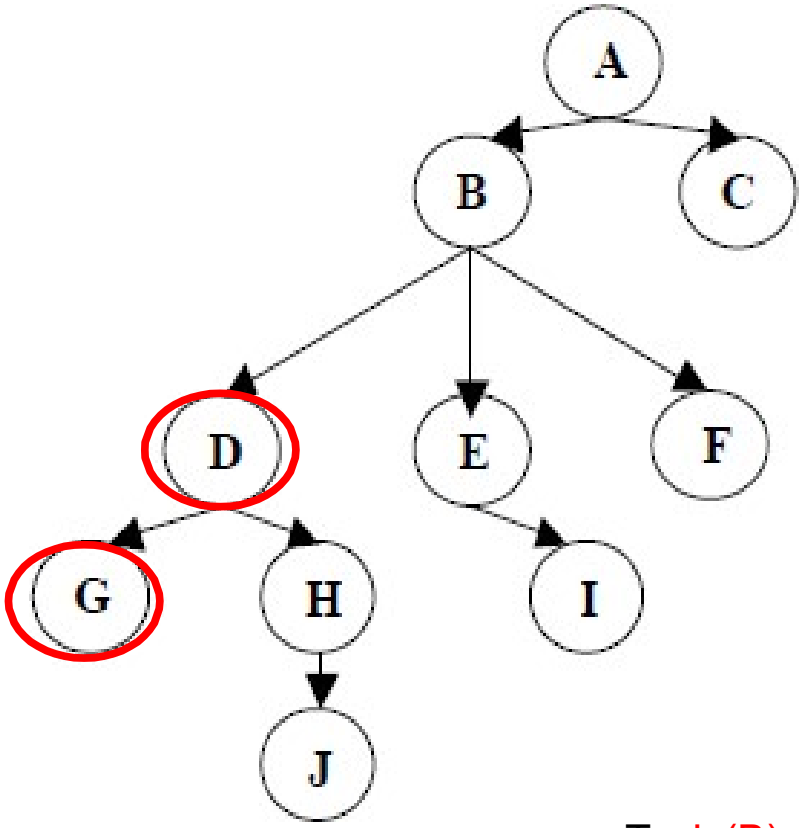
T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



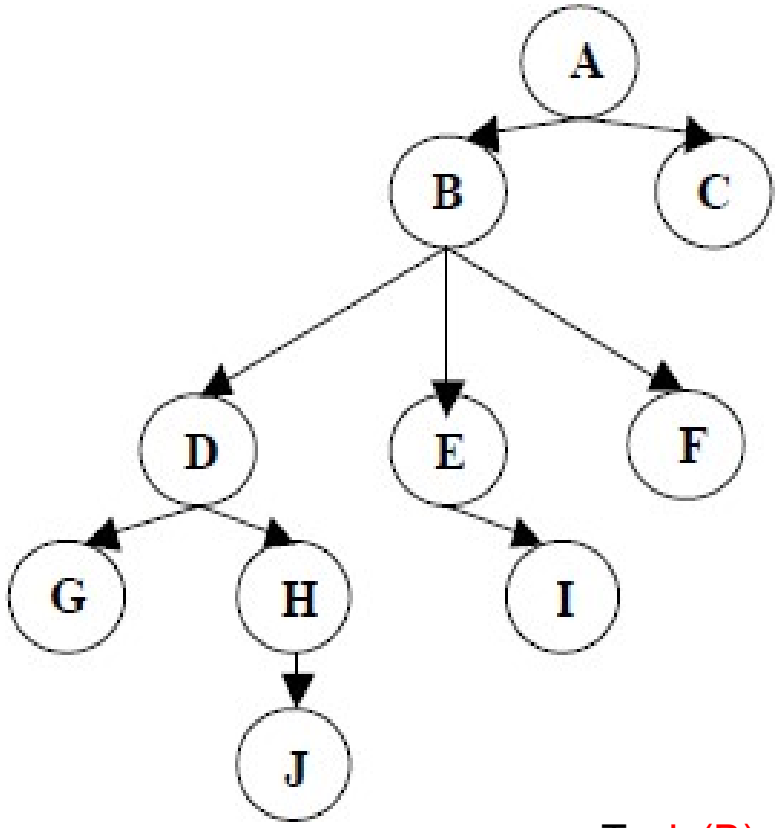
T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



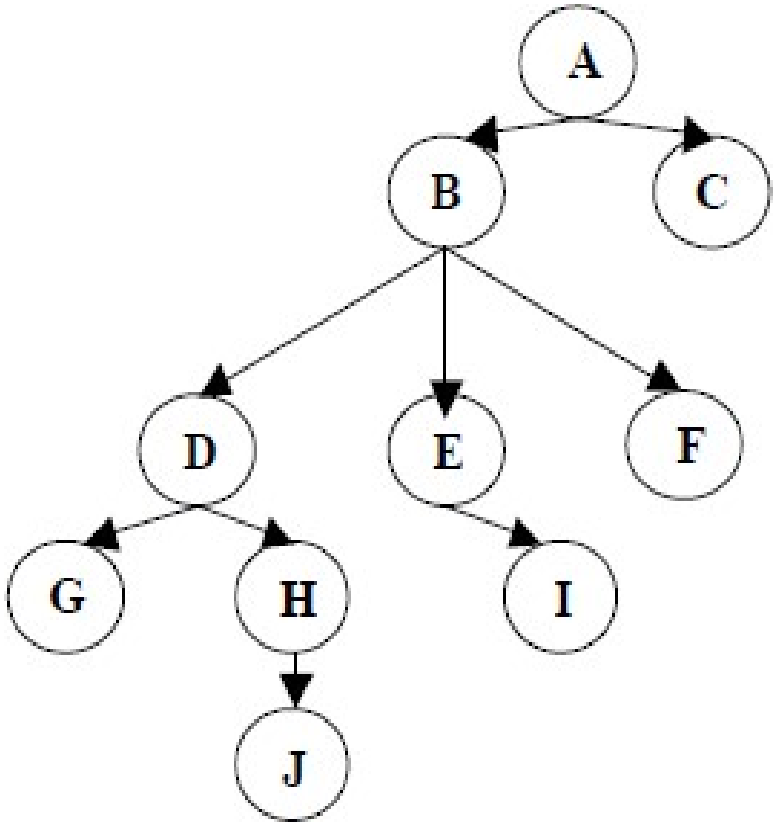
T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



T_1: lx(B), lx(E), ux(E), lx(D), ux(B), lx(G), ux(D), ux(G)

Baseado em Grafo



Exercício:

- A seguinte execução é possível, sob o controle de um protocolo baseado em grafos?

H = lx1(B) lx4(D) lx4(H) ux4(D) lx1(E) ux1(E) lx1(D) ux4(H) ux1(B) lx1(G) ux1(D) ux1(G)

Multigranularidade de Bloqueio

- Visa a redução do número de bloqueios sendo mantidos, a cada momento.

  - Cada item de dado -> tratado individualmente, como uma unidade de sincronização

  - Necessidade de agrupar diversos itens de dados e tratá-los como uma unidade de sincronização individual

Multigranularidade de Bloqueio

Exemplo 1:

- Ti precisa bloquear o BD inteiro
  - Bloqueia cada item do banco (demorado)

    ou

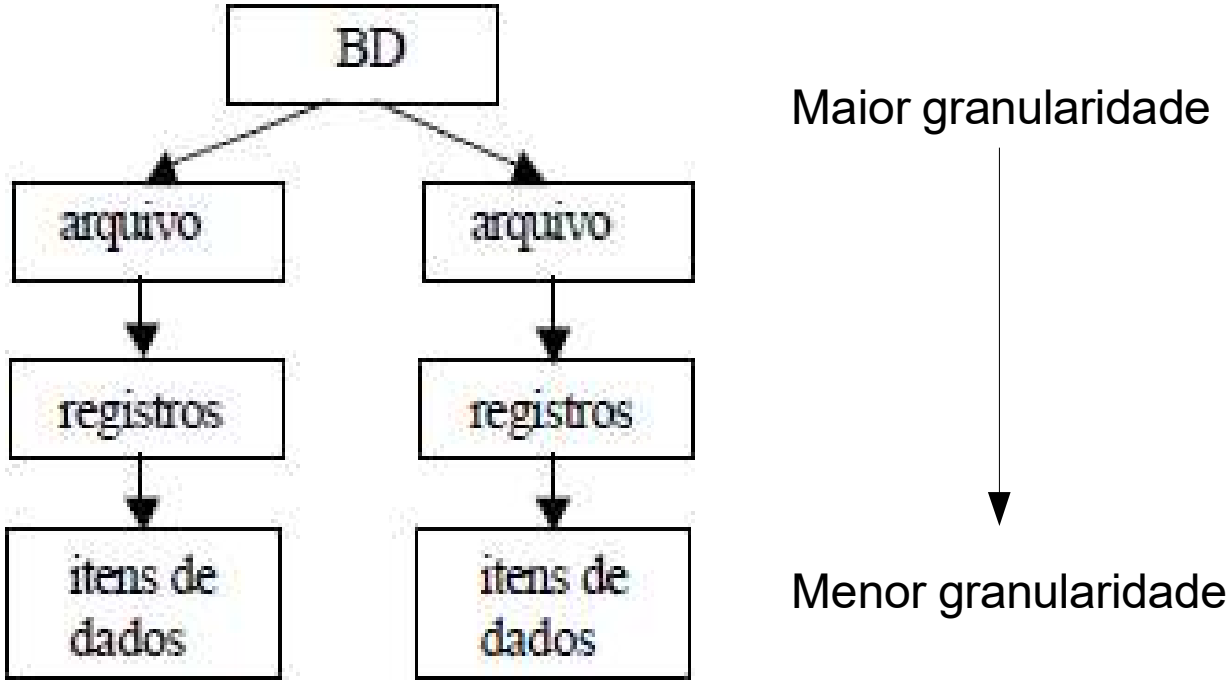
  - Única solicitação bloqueia todo o BD

Exemplo 2:

- Ti precisa acessar apenas uns itens
  - Não precisa bloquear todo o BD (perda de concorrência)
  - Bloqueia o que precisa

Multigranularidade de Bloqueio

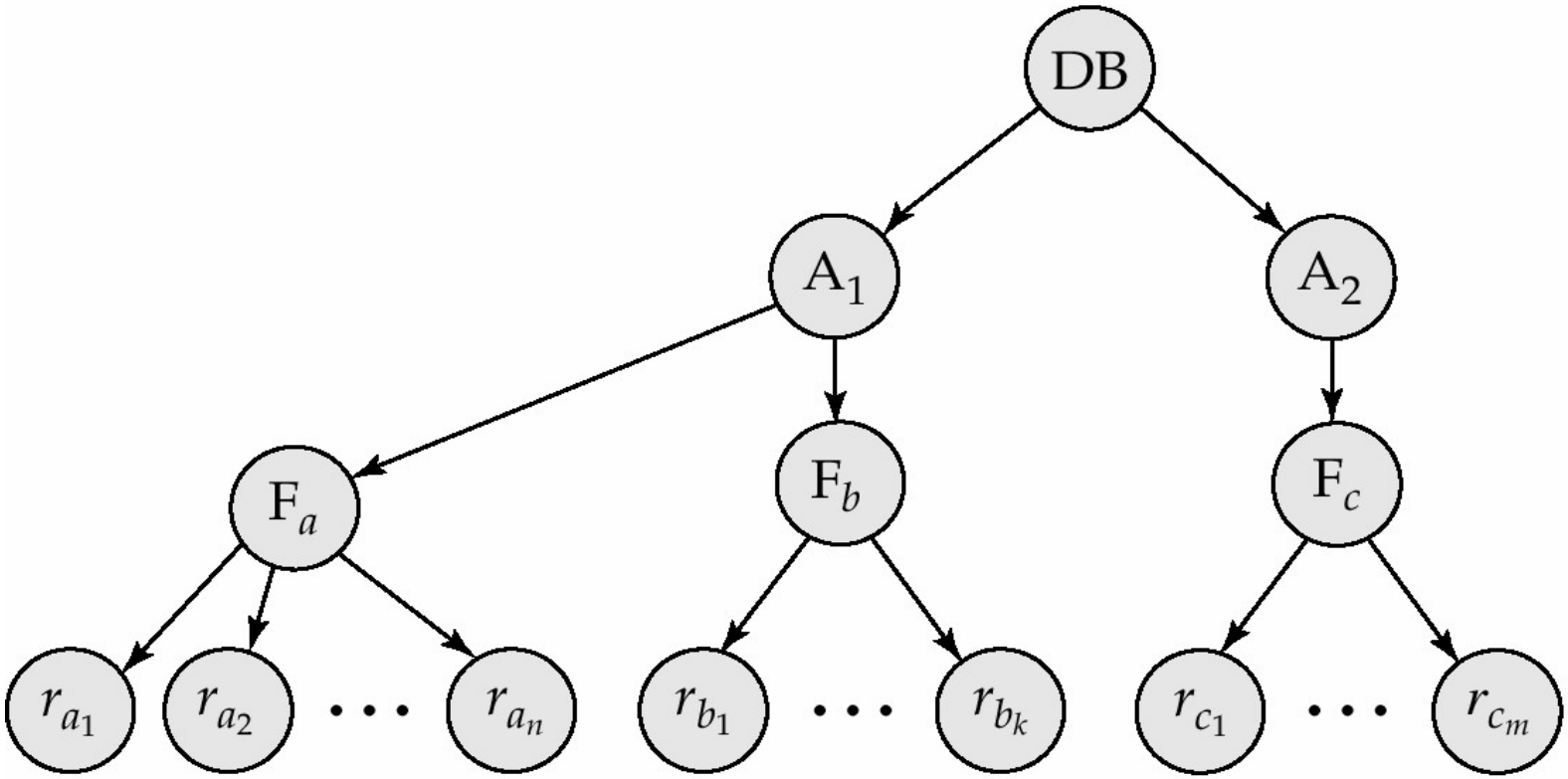
- definir múltiplos níveis de granularidade (tamanho de um item de dado);
- subdividir o BD em níveis para construir uma hierarquia;
- Representação em forma de árvore:



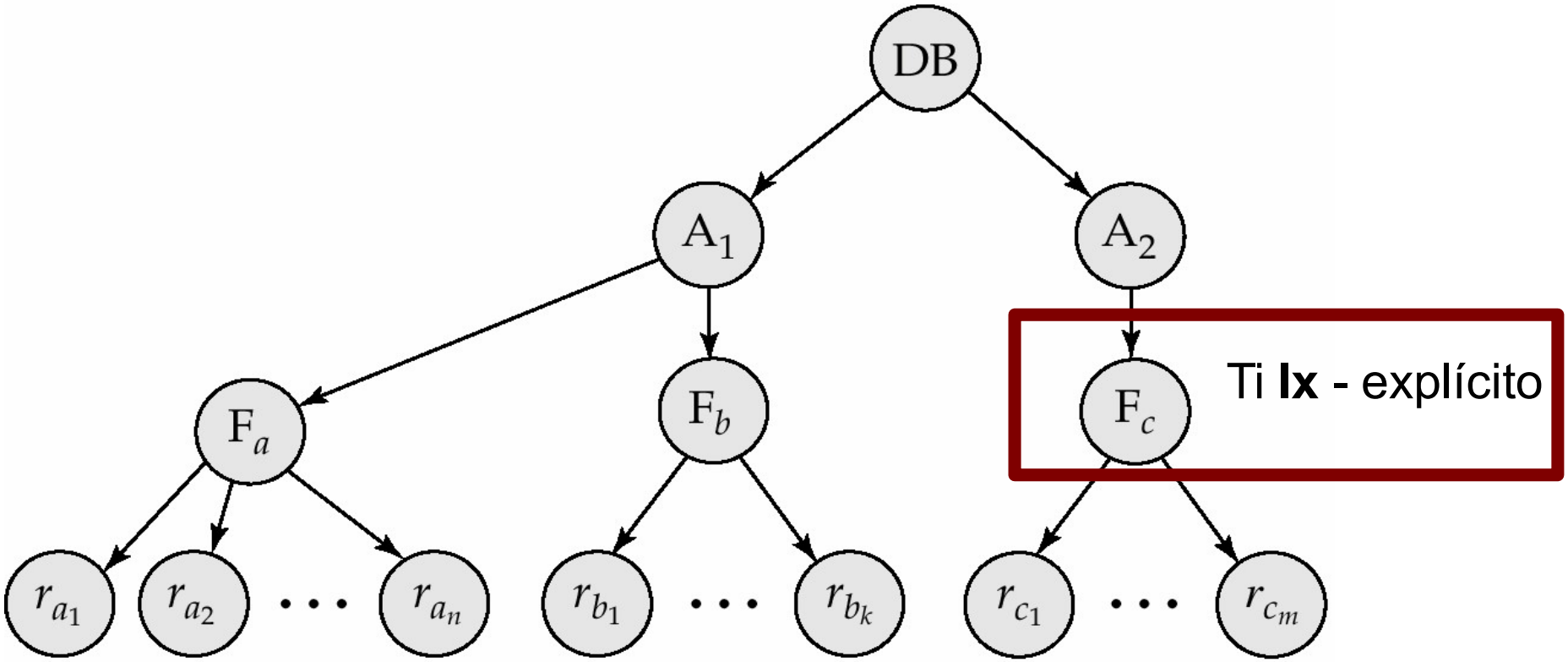
Multigranularidade de Bloqueio

- Tipos de Bloqueio:
  - Partilhado (S)
  - Exclusivo (X)
- Cada nodo é bloqueado individualmente
- Quando um nodo é bloqueado explicitamente, todos os descendentes deste nodo são bloqueados implicitamente (automático)

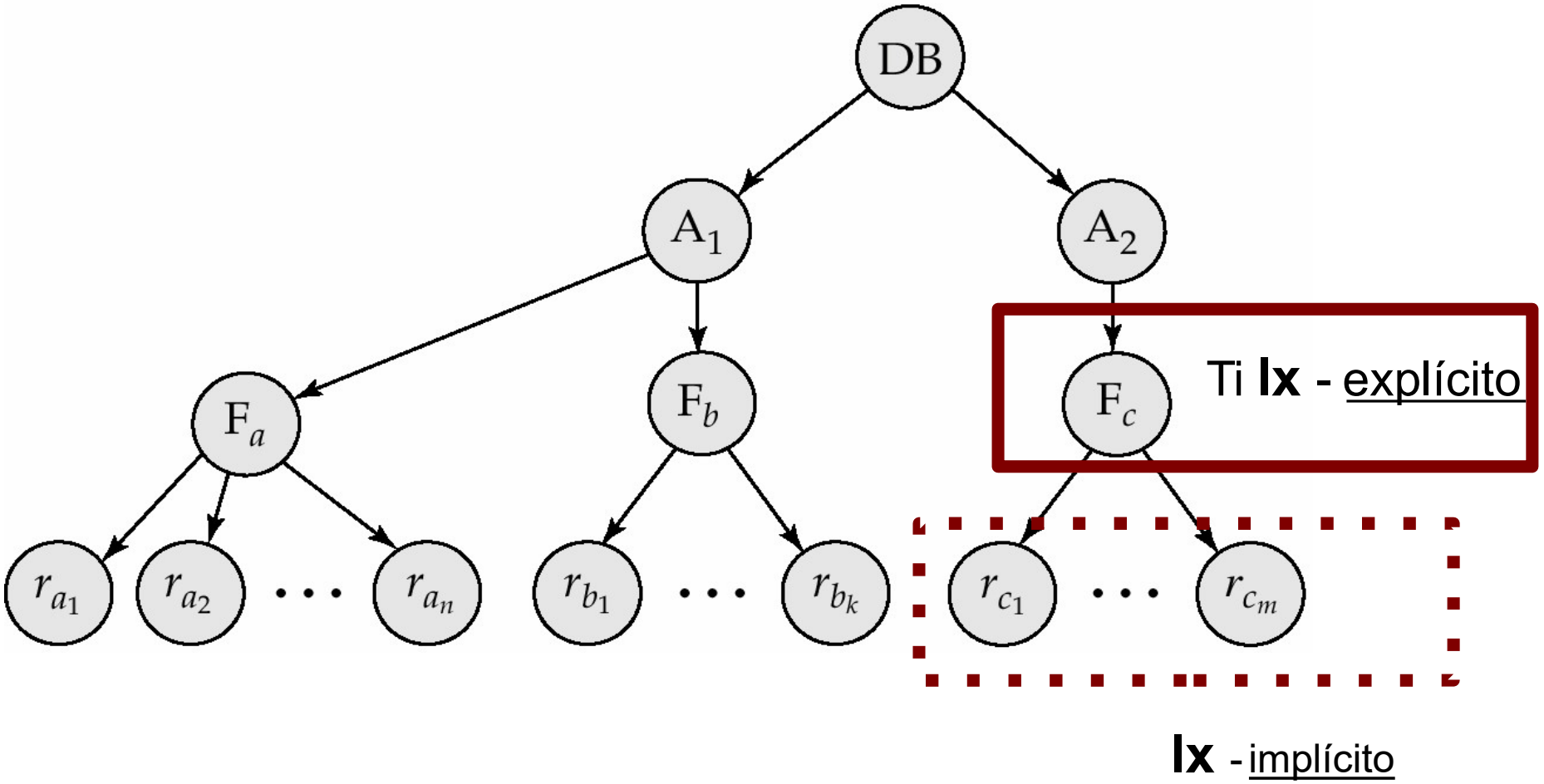
Multigranularidade de Bloqueio



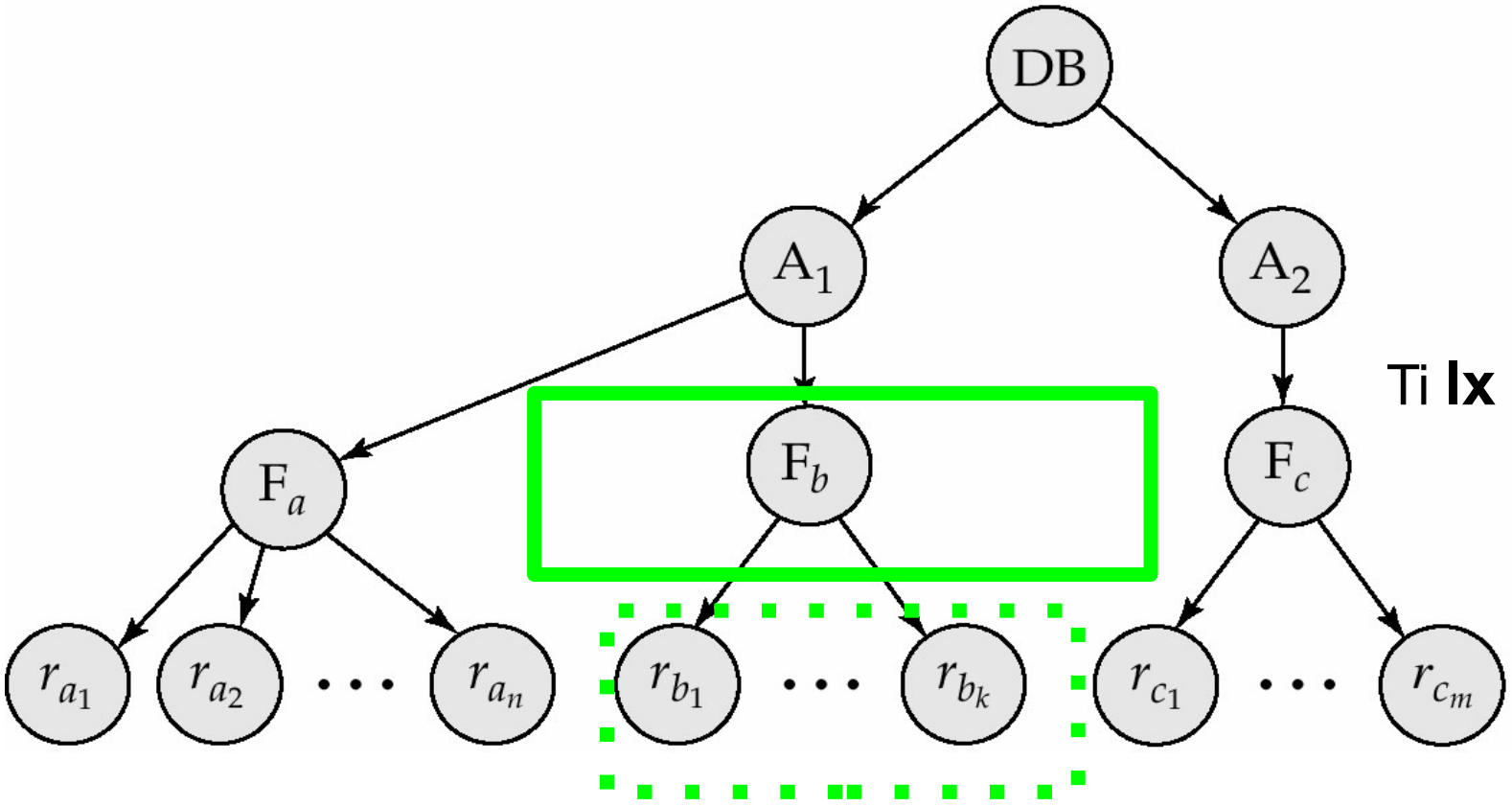
Multigranularidade de Bloqueio



Multigranularidade de Bloqueio



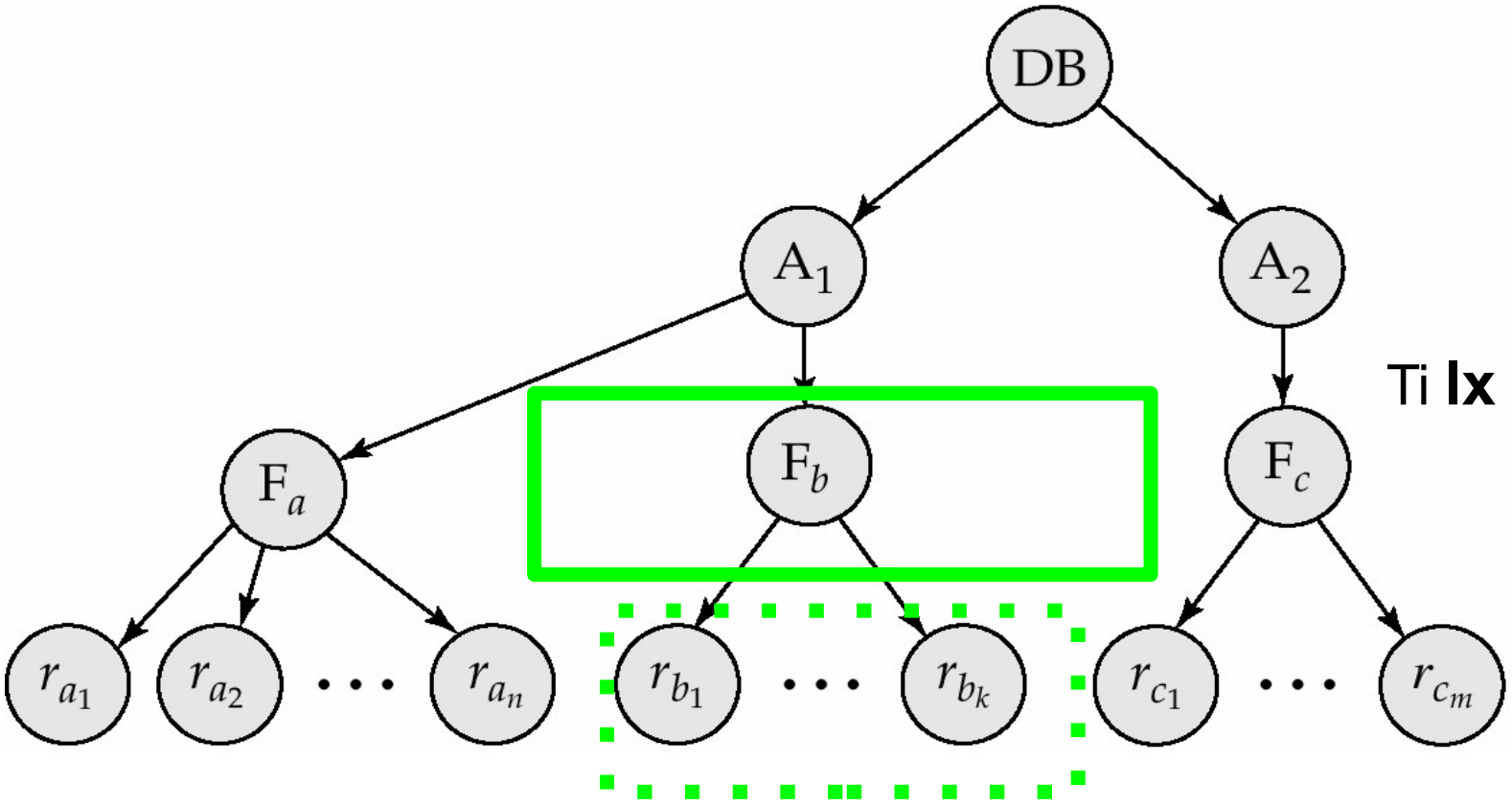
Multigranularidade de Bloqueio



Ti **Ix** - explícito

Ix - implícito

Multigranularidade de Bloqueio

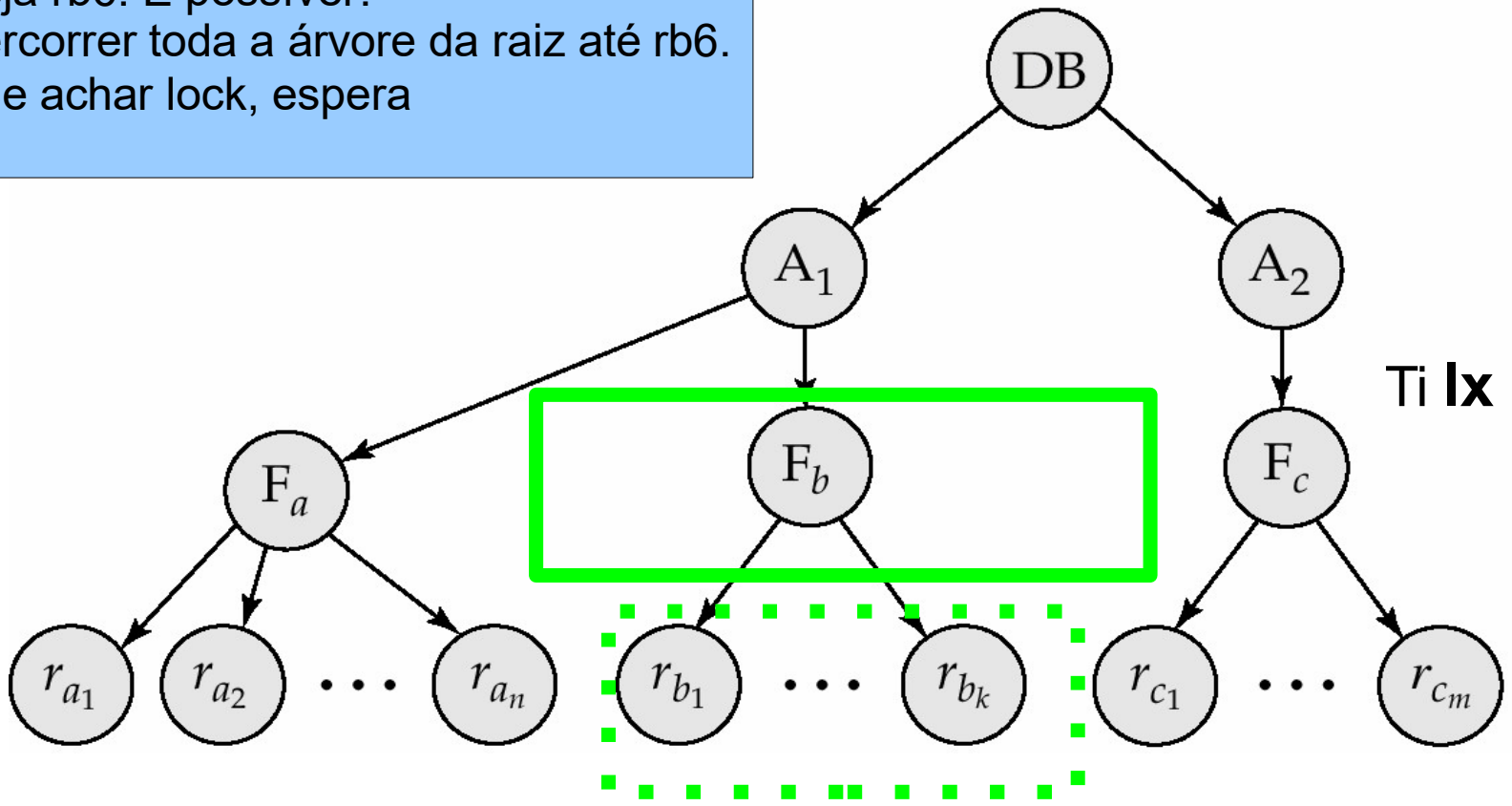


Ti **Ix** - explícito

Ix - implícito

Multigranularidade de Bloqueio

- Tj deseja rb6. É possível?
  - Percorrer toda a árvore da raiz até rb6.
    - Se achar lock, espera



Ti **Ix** - explícito

Ix - implícito

Multigranularidade de Bloqueio

- Questão
  - Uma transação precisa bloquear o BD inteiro. Implicitamente, toda a árvore é bloqueada
  - Se existir um bloqueio em algum registro por outra transação?
    - **Solução 1**: pesquisar toda a árvore antes (desempenho)
    - **Solução 2**: Bloqueio de Intenção

Multigranularidade de Bloqueio

- **Bloqueio de Intenção**

  - Visa evitar conflitos de acesso em granularidades diferentes

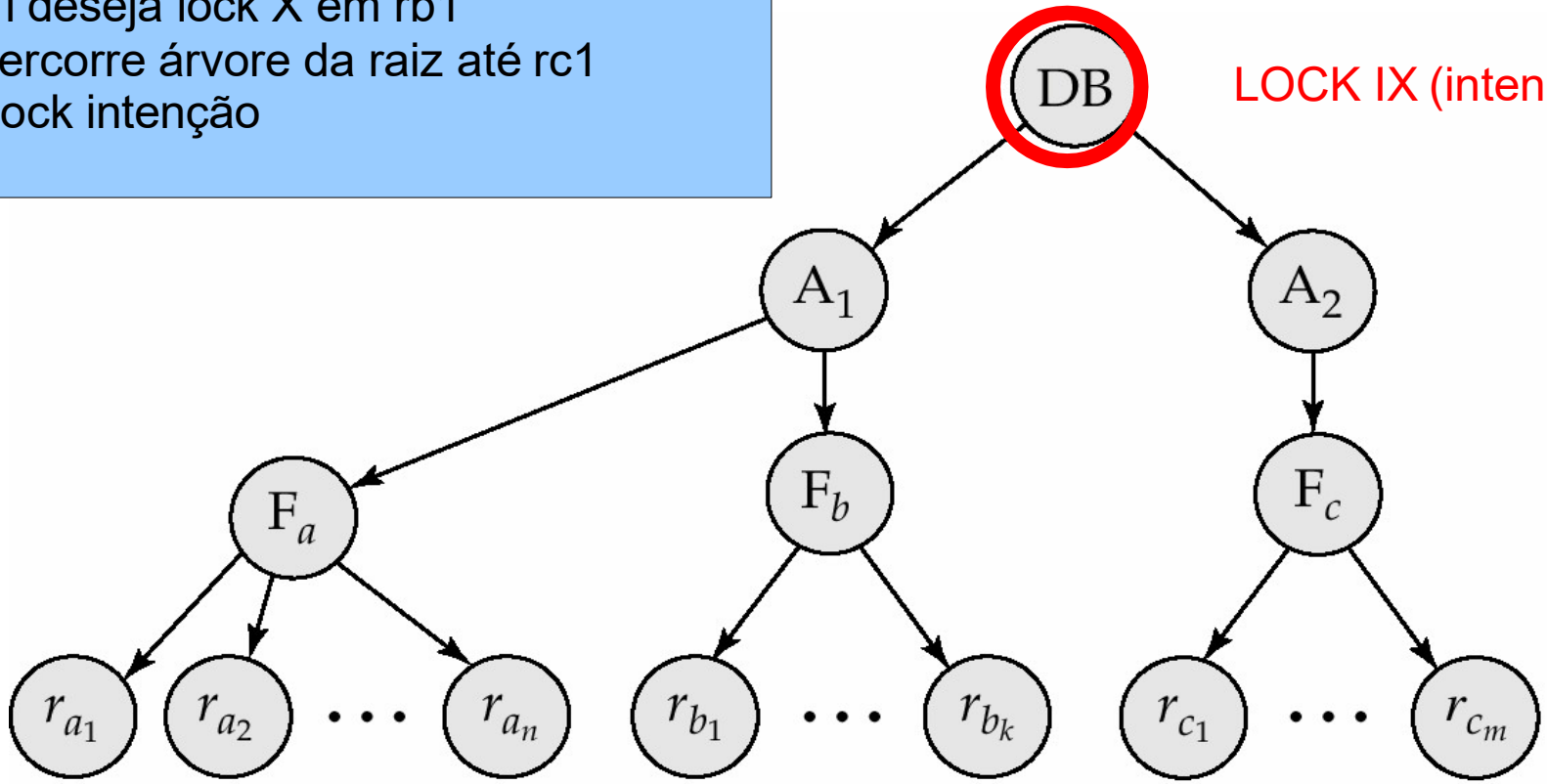
    Significado: sinaliza que existe um bloqueio explícito em um nível inferior da hierarquia

  - Finalidade: Ti não precisa percorrer toda a árvore para verificar existência de bloqueio;

** Os bloqueios de intenção são colocados implicitamente em todos os ancestrais de um nodo antes dele ser bloqueado explicitamente.*

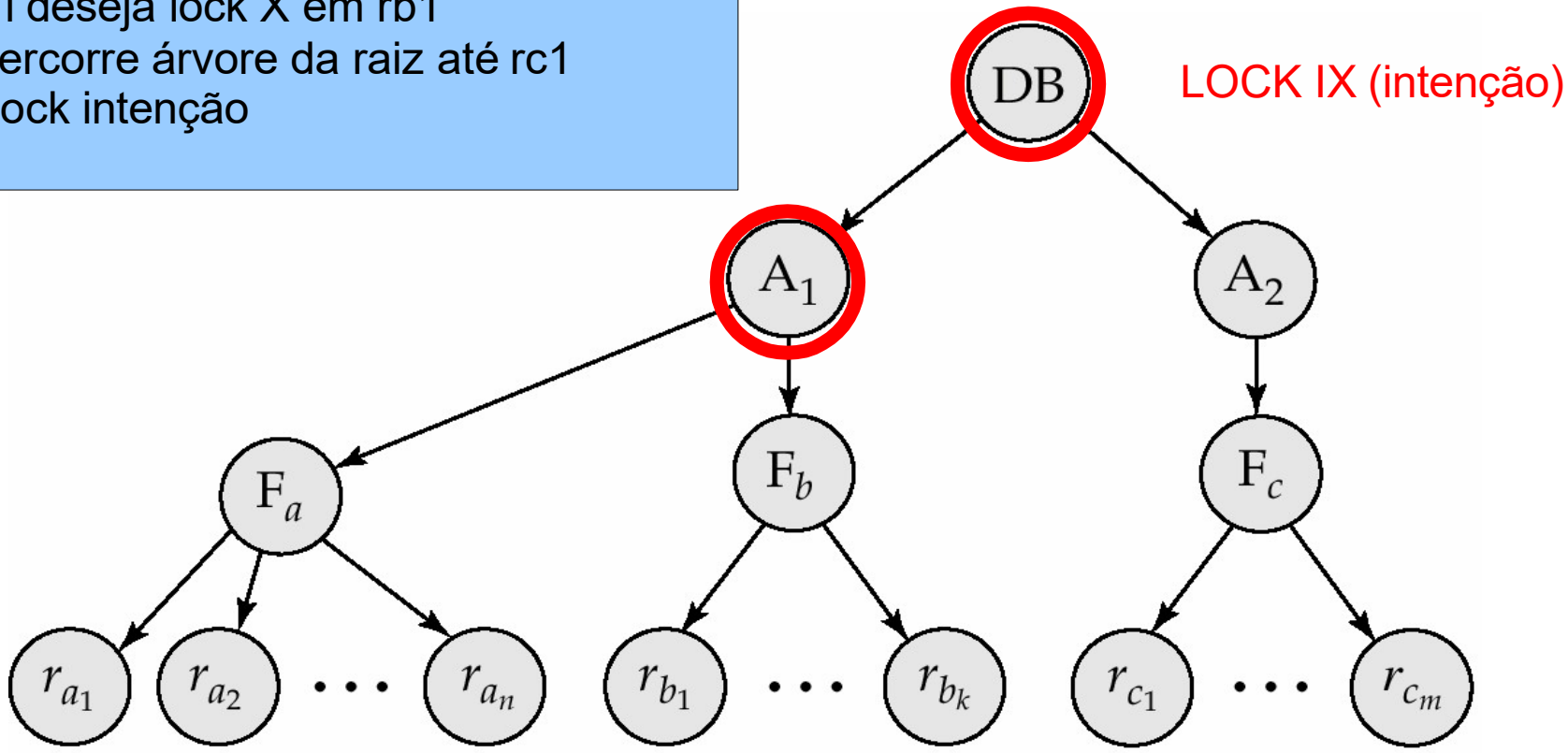
Multigranularidade de Bloqueio

- Ti deseja lock X em rb1
-Percorre árvore da raiz até rc1
e lock intenção



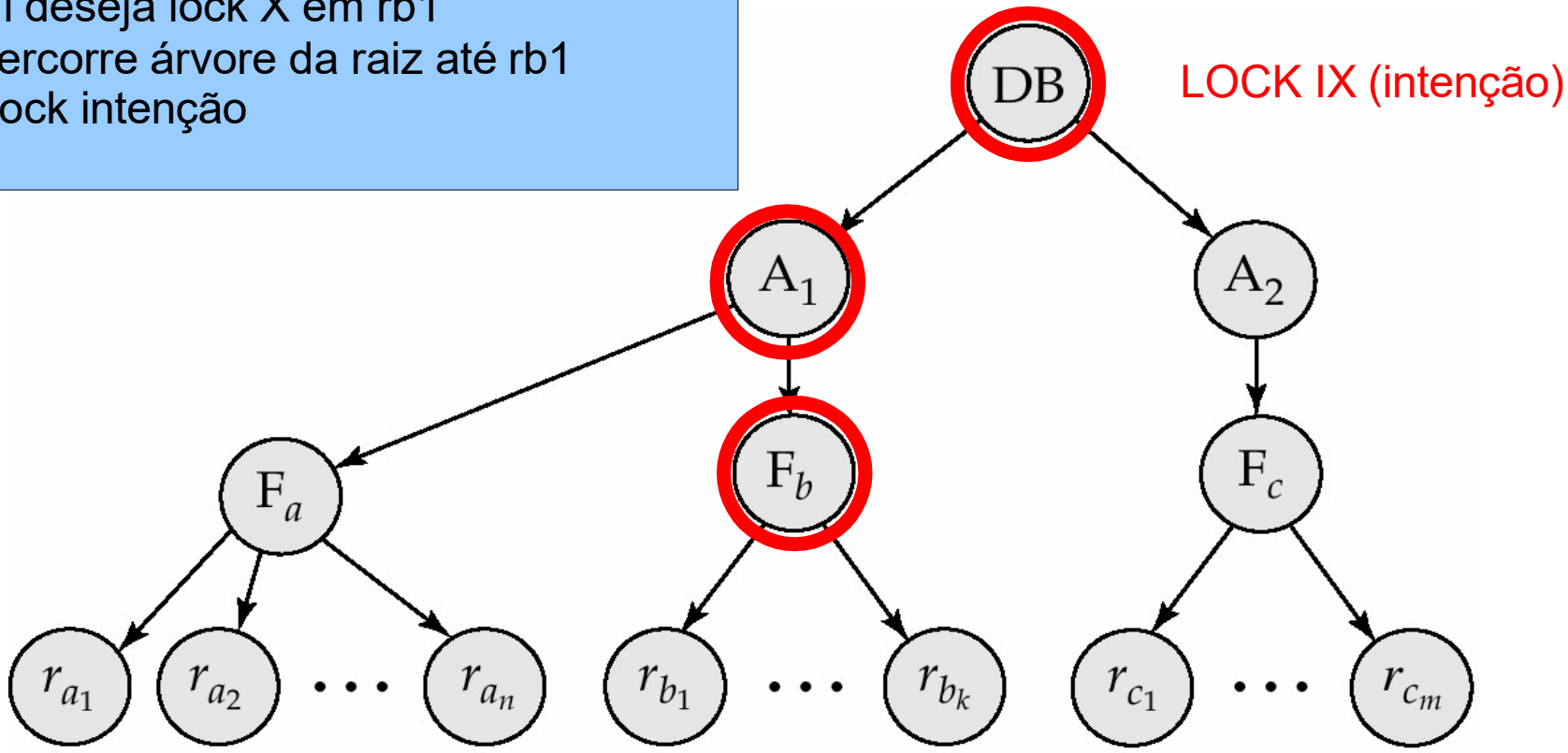
Multigranularidade de Bloqueio

- Ti deseja lock X em rb1
-Percorre árvore da raiz até rc1
e lock intenção



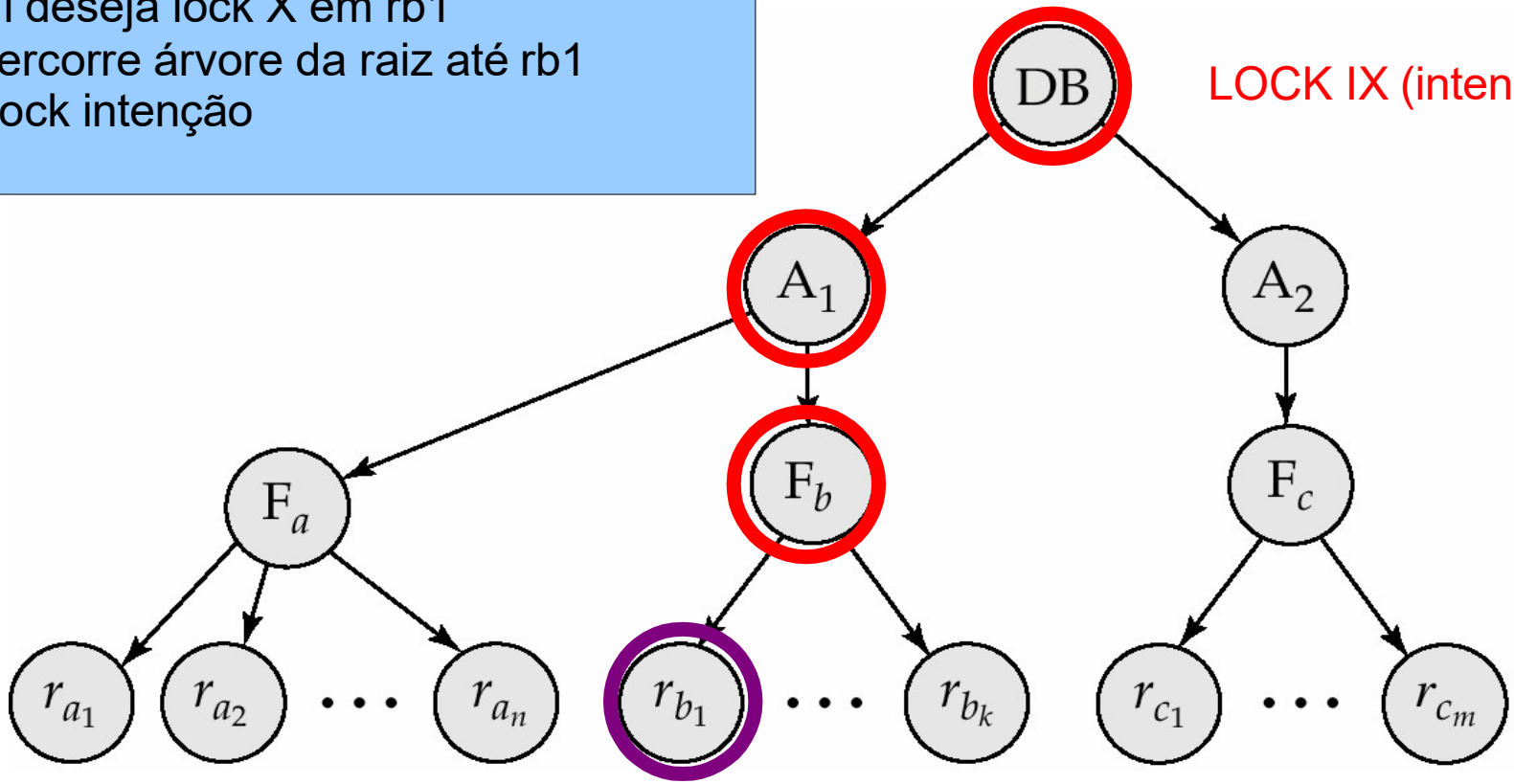
Multigranularidade de Bloqueio

- Ti deseja lock X em rb1
-Percorre árvore da raiz até rb1
e lock intenção



Multigranularidade de Bloqueio

- Ti deseja lock X em rb1
-Percorre árvore da raiz até rb1
e lock intenção



Multigranularidade de Bloqueio

- Modos de Bloqueio de Intenção:
  - **Intenção Partilhado (IS):** indica que existe um bloqueio explícito, no nível mais baixo da árvore, no modo partilhado;
  - **Intenção Exclusivo (IX)**: indica que existe um bloqueio explícito, no nível mais baixo da árvore, no modo exclusivo ou partilhado;
  - **Intenção Partilhado e Exclusivo (SIX):** indica que existe uma subárvore com bloqueio partilhado e um dos níveis inferiores com bloqueios exclusivos;

Multigranularidade de Bloqueio

- Para uma Ti bloquear um nó:
  - 1) Verificar matriz de compatibilidade

bloqueio solicitado	tipo atual de bloqueio				
	ls	lx	is	ix	six
ls	y	n	y	n	n
lx	n	n	n	n	n
is	y	n	y	y	y
ix	n	n	y	y	n
six	n	n	y	n	n

Multigranularidade de Bloqueio

2)A raiz da árvore precisa ser bloqueada primeiro, entretanto, pode ser desbloqueada em qualquer modo;

3)Uma transação pode bloquear um nodo no modo S ou IS somente se os ascendentes estiverem bloqueados pela mesma transação nos modos IX ou IS;

4)Uma transação pode bloquear um nodo no modo X, SIX ou IX somente se os ascendentes estiverem bloqueados pela mesma transação nos modos IX ou IS;

5)As regras do protocolo de Bloqueio de Duas Fases devem ser seguidas – uma transação só pode realizar bloqueios enquanto não realiza nenhum desbloqueio;

Multigranularidade de Bloqueio

6) Um nodo pode ser desbloqueado somente se seus descendentes já foram desbloqueados anteriormente;

- Bloqueios: sentido TOP-DOWN - raiz para as folhas (primeiro bloqueios de intenção, depois bloqueios);
- Desbloqueios: sentido BOTTOM-UP - folhas para raiz (primeiro bloqueios, depois bloqueios de intenção).

Multigranularidade de Bloqueio

- Conclusões:
  - aumento de concorrência;
  - reduz sobrecarga de bloqueios.
- - Utilidades:
  - transações pequenas que utilizam poucos itens de dados;
  - transações longas que produzem relatórios de um arquivo inteiro ou de um conjunto de arquivos.
- - Comentários:
  - garante a serializabilidade
  - não evita deadlock

Marcador de Tempo

- Realiza ordenação das transações para que as operações conflitantes apareçam serializáveis
- O sistema associa a cada transação Ti, antes dela iniciar sua execução, um marcador de tempo (Ts[Ti])
  - Se T1 tem o marcador Ts[T1] e uma nova transação Tj entrar no sistema: Ts(Ti) <Ts(Tj)

Uso Relógio do Sistema ou Contador cronológico

Marcador de Tempo

- Funcionamento:
  - Para cada item de dado (x) associa-se:
    - **W-Ts (x):** maior marcador de tempo de qualquer Tn que executa write com sucesso.

      R_Ts (x): maior marcador de tempo de qualquer Tn que executa read com sucesso.
  - Atualiza-se os marcadores com a execução de operações read e write.

Marcador de Tempo

- Operações de Leitura em Ti:
  - Se TS(Ti) < W-TS(x):
    - operação read rejeitada;
    - Ti refeita.
  - Se TS(Tj) >= W-TS(x):
    - operação read executada;
    - R-TS(x) ajustado.
- Operações de Escrita em Ti:
  - Se TS(Ti) < R-TS(x):
    - operação write rejeitada;
    - Ti refeita.
  - Se TS(Tj) < W-TS(x):
    - operação write rejeitada;
    - Ti refeita.
  - Caso contrário, executa write e W-TS(x) é ajustado.

Marcador de Tempo - Exemplo

- A execução abaixo segue o protocolo de marcador de tempo?
  r1(b)->r2(b)->w2(b)->r1(a)->r2(a)->w2(a)

1	2
r(b)	
	r(b)
	w(b)
r(a)	
	r(a)
	w(a)

TS1:
r(b)
w(b)
r(a)
w(a)

TS2:

Se TS(Ti) < W-TS(x):

- operação read rejeitada;

Se TS(Tj) >= W-TS(x):

- operação read executada;

Se TS(Ti) < R-TS(x):

- operação write rejeitada

Se TS(Tj) < W-TS(x):

- operação write rejeitada;