Drinking Philosophers

Algoritmos Distribuídos

Professora: Lúcia Drummond

Drinking Philosophers

Os nós podem acessar diferentes subconjuntos de recursos sempre que eles exigem acesso a recursos compartilhados, então passa a existir a possibilidade de que vizinhos em G acessem recursos compartilhados concorrentemente.

A técnica de empregar um único garfo por aresta para assegurar acesso exclusivo aos recursos que dois vizinhos compartilham não é mais suficiente.

Drinking Philosophers

Ao invés disso, associado com todas as arestas deve existir um objeto para cada recurso que os correspondentes vizinhos compartilham.

Tais objetos são garrafas.

Algoritmo Drinking _Philosophers

- **Variáveis**

  $thirsty_i$ = **false**;

  $holds\_bottle_i^{jk}$ = **false** para todo $n_j \in Neig_i$ e todo $b_k \in B_i^j$;

  $holds\_turn_i^j$ = **false** para todo $n_j \in Neig_i$;

  $owes\_bottle_i^{jk}$ = **false** para todo $n_j \in Neig_i$ e todo $b_k \in B_i^j$;

  $needs\_bottle_i^{jk}$ = **false** para todo $n_j \in Neig_i$ e todo $b_k \in B_i^j$;

  X_i=0;

  Y_i=0;

Algoritmo Drinking _Philosophers

- **Algoritmo**

(1)Input:

msg_i = **nil**;

Ação quando not *thirsty* e necessita-se do acesso ao recurso compartilhado:

$thirsty_i$:= **true**;

$needs\_bottle_i^{jk}$:= **true** para todo $n_j \in Neig_i$ e todo $b_k \in B_i^j$ tal que acesso ao recurso que b_k representa é necessário;

Algoritmo Drinking _Philosophers

para todo $n_j \in Neig_i$ tal que existe $b_k \in B_i^j$ com *needs_bootle*$_i^{jk}$=**true e** *holds_bottle*$_i^{jk}$=**false do**

begin

Seja X_i o subconjuntos de B_i^j tal que $b_k \in X_i$ se e somente se *needs_bottle*$_i^{jk}$=**true** e *holds_bottle*$_i^{jk}$=**false;**

Envie *request*(X_i) para todo n_j;

$X_{i;}$=0;

end

Algoritmo Drinking _Philosophers

- **Algoritmo (cont)**

(2)Input:

msg_i = *request*(X) tal que $origem_i(msg_i)=n_j$;

Ação:

para todo $b_k \in X$ **do**

if not $thirsty_i$ **or not** $holds\_turn_i^j$ **or not** $needs\_bottle_i^{jk}$ **then**

begin

$holds\_bottle_i^{jk}$:=**false;**

$X_i := X_i \cup \{b_k\}$;

if *thirsty* **and** $needs\_bottle_i^{jk}$ **then**

$\mathbf{Y}_i := Y_i \cup \{b_k\}$;

end

Algoritmo Drinking _Philosophers

```
    else
        owes_bottle_i^jk := true;
if X_i ≠ 0 then
    begin
        if Y_i = 0 then
            Envie bottle(X_i,nil) para n_j;
        else
            begin
                Envie bottle(X_i,request(Y_i)) para n_j;
                Y_i := 0;
            end
        X_i := 0;
    end
```

Algoritmo Drinking _Philosophers

- **Algoritmo (cont)**

(3)Input:

msg_i = *bottle*(X,t) tal que $origem_i(msg_i)=n_j$;

Ação:

$holds\_bottle_i^{jk}$:= **true** para todo $b_k \in X$;

if t = *turn* **then** $holds\_turn_i^j$:= **true;**

if t = *request*(Y) **then**

$owes\_bottle_i^{jk}$:=**true** para todo $b_k \in Y$;

if $holds\_bottle_i^{kl}$ para todo $n_k \in Neig_i$ e todo $b_l \in B_i^k$ **then**

Algoritmo Drinking _Philosophers

```
begin
    acessa recursos compartilhados;
    thirsty_i := false;
    para todo n_k ∈ Neig_i do
        if holds_turn_i^k then
            begin
                holds_turn_i^k := false;
                para todo b_l ∈ B_i^k do
                    if owes_bottle_i^kl then
                        begin
                            owes_bottle_i^kl := false;
                            holds_bottle_i^kl := false;
                                X_i := X_i ∪ {b_l};
                        end
```

Algoritmo Drinking _Philosophers

if $X_i \neq 0$ **then**
begin
Envie *bottle*(X_i,*turn*)
para n_k;
X_i:=0;
end
else
Envie *turn* para n_k;
end
end

Algoritmo Drinking _Philosophers

- **Algoritmo (cont)**

(4)Input:

msg_i = *turn* tal que $origem_i(msg_i)=n_j$;

Ação:

$holds\_turn_i^j$:= **true;**