CS586: Distributed Computing

Tutorial 5

Professor: Panagiota Fatourou
TA: Eleftherios Kosmas
CSD - December 2011

Snapshots - T-Opt

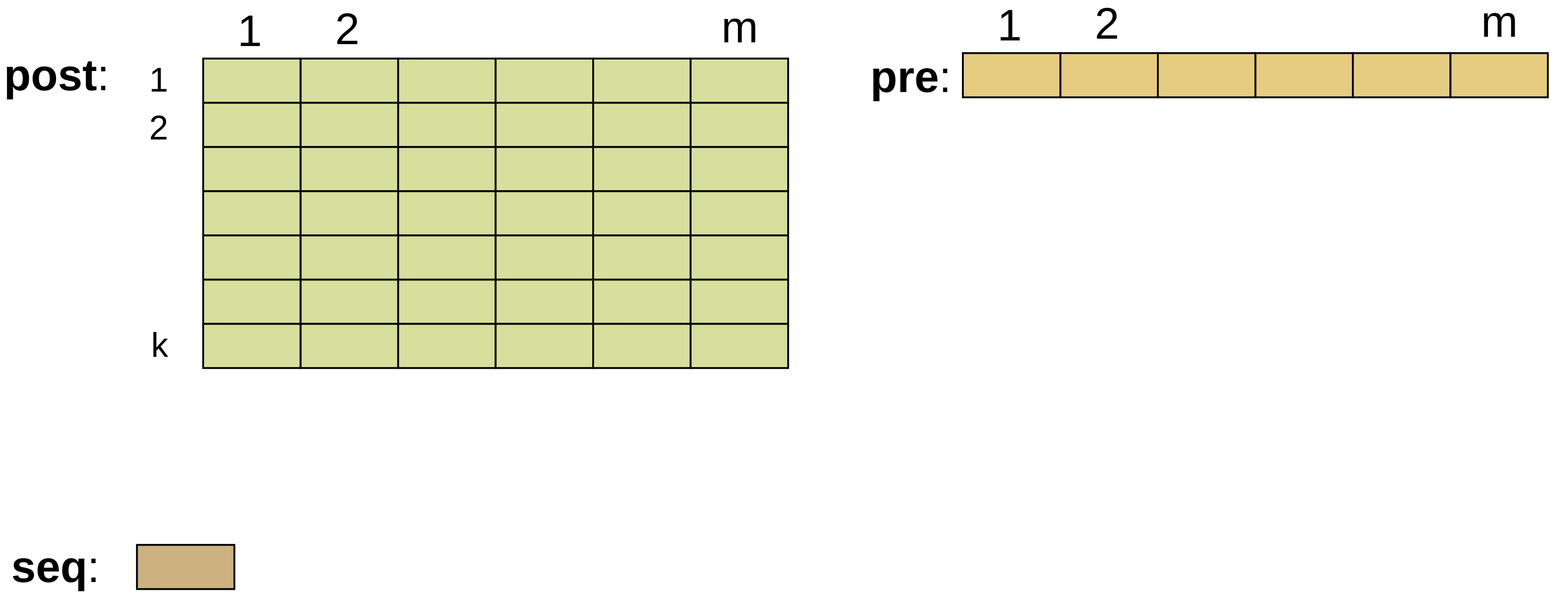
```
void update (data value, int i) {
    int curr_seq;
    data d1, d2;
    curr_seq = seq;
    d1 = pre[i];
    d2 = post[curr seq][i];
    if (d2 == null)
        post[curr seq][i]=d1;
    pre[i]=value;
}
```

Initially

seq = 1
post[1..k][1..m] = {null, null, …, null}
pre[1..m] = {null, null, …, null}

```
data *scan (void) {
    data view[1..m], d1, d2;
    int j;
    seq=seq+1;
    for (j = 1; j ≤ m; j++) {
        d1 = pre[j];
        d2 = post[seq][j];
        if(d2 == null) view[j]=d1;
        else view[j]=d2;
    }
    return view;
}
```

Snapshots - T-Opt



T-Opt - Linearizability

- α is an execution of T-opt and **S** is any SCAN performed in α
  - **w_S** : is the write performed by S (line 7)
  - **seq_S** : is the value written to seq by w_S
- For each $i \in \{1,\ldots, m\}$,
  - **r_i^S** : the read of pre[i] by S (line 9)
  - **$\tilde{}r_i^S$** : the read of post[seq_S][i] by S (line 10)
  - **v_i** : the value that S returns for component A_i
  - If S reads null at $\tilde{}r_i^S$ and v_i at r_i^S
    - **U_i^S** : the UPDATE that writes v_i to pre[i] and its write to it is the last write to it that precedes r_i^S
  - If S reads v_i at $\tilde{}r_i^S$
    - **V_i^S** : the UPDATE that writes v_i to post[seq_S][i] and its write to it is the last write to it that precedes $\tilde{}r_i^S$
    - **U_i^S** : the UPDATE that writes v_i to pre[i] and its write to it is the last write to it before V_i^S reads pre[i]
  - **w_i^S** : the write to pre[i] by U_i^S (line 6)

Snapshots - T-Opt

```
void update (data value, int i) {
    int curr_seq;
    data d1, d2;
    curr_seq = seq;
    d1 = pre[i];
    d2 = post[curr seq][i];
    if (d2 == null)
        post[curr seq][i]=d1;
    pre[i]=value;                // w_i^S
}
```

```
data *scan (void) {
    data view[1..m], d1, d2;
    int j;
    seq=seq+1;                   // w_S
    for (j = 1; j ≤ m; j++) {
        d1 = pre[j];             // r_i^S
        d2 = post[seq][j];       // ~r_i^S
        if(d2 == null) view[j]=d1;
        else view[j]=d2;
    }
    return view;
}
```

Initially
seq = 1 post[1..k][1..m] = {null, null, …, null} pre[1..m] = {null, null, …, null}

T-Opt - Linearization points

- Each SCAN S is linearized at w_S
- For each $i \in \{1,\ldots, m\}$,
  - if w_i^S follow w_S,
    - U_i^S is linearized **just before w_S**
    - each UPDATE on A_i that performs its write to pre[i] between w_S and w_i^S is linearized **just before w_S**
      - ties are broken by the order that the writes to pre[i] occur
  - each of the rest of UPDATES is linearized at its **write to pre[i]** (line 6)

T-Opt - Linearizability: Intuition

T-Opt is **linearizable**

A. The linearization point of each operation is **within its execution interval**

Intuition:

- SCANs?
- UPDATEs linearized when they write to pre[i]?
- UPDATEs linearized before the linearization point of some SCAN?

B. Scans returns **consistent vectors**

Intuition: (v_i is written by the last UPDATE linearized before SCAN)

- if w_i^S follows w_S?
- if w_i^S precedes w_S?

hint: the linearization order of UPDATEs on A_i respects the order of writes to pre[i] (by those UPDATEs)

T-Opt - Linearizability: sketch of proof

T-Opt is **linearizable**

A. The linearization point of each operation is **within its execution interval**

A.1. If w_i^S follows w_S, then the execution of an UPDATE that performs its write to pre[i] between w_S and w_i^S (including U_i^S) starts before w_S

B. Scans returns **consistent vectors**

B.1. The linearization order of the UPDATES on any component A_i respects the order in which these UPDATES perform their writes to pre[i]

T-Opt - Technical Lemmas

Lemma 1. For each $i \in \{1,\ldots,m\}$, **$\tilde{}r_i^S$ follows w_i^S**

sketch of proof

- if w_i^S precedes w_S …
- if w_i^S follows w_S
  - Assume, S reads null at $\tilde{}r_i^S$ and v_i at r_i^S …
  - Assume, S reads v_i at $\tilde{}r_i^S$ …

Lemma 2. Assume that S reads v_i at $\tilde{}r_i^S$, and let r_{pre} be the read of pre[i] by V_i^S. Then, **r_{pre} follows w_S**

sketch of proof

- Assume, by the way of contradiction, that r_{pre} is executed before w_S
- Then, the read of seq by V_i^S precedes w_S …

T-Opt

Lemma A.1. For each $i \in \{1,\ldots,m\}$, such that w_i^S follows w_S, it holds that any UPDATE on A_i that performs its write to pre[i] between w_S and w_i^S (including U_i^S) begins its execution before w_S

sketch of proof

- Assume, by the way of contradiction, that there is an UPDATE U on A_i that starts its execution after w_S and performs its write to pre[i] (let it be w) before w_i^S
- U reads seq_S in seq
  - Lemma 1 → U ends its execution before the end of S
  - U starts after w_S
- S read a value other than null at $\tilde{}r_i^S$
  - U executes lines 4-5 before w (which precedes w_i^S)
- V_i^S reads a value other than null in post[seq_S][i]

T-Opt

Lemma A. Let α be any execution of T-Opt. The linearization point of any SCAN or UPDATE executed in α is within its execution interval

sketch of proof

- SCANs
- UPDATEs linearized at their writes to pre

- Let U be an update on A_i which is not linearized at its write to pre[i]
- There is a SCAN S such that
  - w_i^S is executed after w_S
  - the write to pre[i] by U is executed between w_S and w_i^S
  - U is linearized just before w_S
- Lemma A.1. → U begins its execution before w_S

T-Opt

Lemma B.1. Let U_1, U_2 be two update on some component A_i, $1 \leq i \leq m$. Denote by w_1 the write to pre[i] by U_1 and by w_2 the write to pre[i] by w_2. If w_1 precedes w_2, the linearization point of U_1 precedes the linearization point of U_2

sketch of proof

- Assume, by the way of contradiction, that the claim does not hold
- U_1 and U_2 are linearized at their writes to pre[i]…
- At least one of the U_1, U_2 is not linearized at its write to pre[i]
  - U_2 is linearized at w_2 … (*hint: use Lemma A.*)
  - U_1 is linearized at w_1
    - U_2 can not be linearized at w_2 (why?)
    - Therefore a SCAN S exists such that
      - w_2 has been performed between w_S and w_i^S
    - w_1 precedes w_i^S, since w_1 precedes w_2
    - if w_1 follows w_S …
    - if w_1 precedes w_s … (*hint: use Lemma A.*)

T-Opt

Lemma B.1. Let U_1, U_2 be two update on some component A_i, $1 \leq i \leq m$. Denote by w_1 the write to pre[i] by U_1 and by w_2 the write to pre[i] by w_2. If w_1 precedes w_2, the linearization point of U_1 precedes the linearization point of U_2

sketch of proof

- None of U_1, U_2 is linearized at its write to pre[i]
  - Two SCANS S1 and S2 exist such that
    - w_1 has been performed between w_{S1} and w_i^{S1}
    - w_2 has been performed between w_{S2} and w_i^{S2}
  - if S1 = S2 …
  - if S1 follows S2 …
    - Lemma 1 → w_i^{S2} precedes the end of S2
  - if S1 precedes S2 …
    - Lemma 1 → w_i^{S1} precedes the end of S1

T-Opt

Lemma B. Let α be any execution of T-Opt. Any SCAN executed in α returns a consistent vector

sketch of proof

- Recall that U_i^S store v_i in component A_i and its linearization point precedes the linearization point of S
- Assume, by the way of contradiction, that there is an integer $i \in \{1,\ldots,m\}$ such that the last UPDATE on A_i which has been linearized before S is not U_i^S
  - denote by U this update and let w be the write to pre[i] by U
- w follows w_i^S
  - if w precedes w_i^S, Lemma B.1. implies that U is linearized before U_i^S
- S reads null at $\tilde{r}_i^S$ and v_i at r_i^S
  - U can not be linearized at w
    - w follows r_i^S (why?)
    - U is linearized before S, and S is linearized at w_S

T-Opt

Lemma B. Let α be any execution of T-Opt. Any SCAN executed in α returns a consistent vector

sketch of proof

- S reads null at $\tilde{r}_i^S$ and v_i at r_i^S
  - U can not be linearized at w
    - w follows r_i^S (why?)
    - U is linearized before S, and S is linearized at w_S
  - Therefore, there is a SCAN S' such that
    - w is performed between $w_{S'}$ and $w_i^{S'}$
    - U is linearized just before $w_{S'}$
  - $S \neq S'$, since w follows w_i^S
  - if S' follows S …
  - if S' precedes S …
    - Lemma 1 → $w_i^{S'}$ precedes the end of S'

T-Opt

Lemma B. Let α be any execution of T-Opt. Any SCAN executed in α returns a consistent vector

sketch of proof

- Recall that U_i^S store v_i in component A_i and its linearization point precedes the linearization point of S
- Assume, by the way of contradiction, that there is an integer $i \in \{1,\ldots,m\}$ such that the last UPDATE on A_i which has been linearized before S is not U_i^S
  - denote by U this update and let w be the write to pre[i] by U
- w follows w_i^S
  - ☞ if w precedes w_i^S, Lemma B.1. implies that U is linearized before U_i^S
- S reads v_i at $\tilde{r}_i^S$
  - let r_{pre} be the read of pre[i] by V_i^S

T-Opt

Lemma B. Let α be any execution of T-Opt. Any SCAN executed in α returns a consistent vector

sketch of proof

- S reads v_i at $\tilde{r}_i^S$
  - let r_{pre} be the read of pre[i] by V_i^S
  - U can not be linearized at w
    - w follows r_{pre} (why?)
    - Lemma 2 → r_{pre} follows w_S
    - U is linearized before S, and S is linearized at w_S
  - Therefore, there is a SCAN S' such that
    - w is performed between $w_{S'}$ and $w_i^{S'}$
    - U is linearized just before $w_{S'}$
  - $S \neq S'$, since w follows w_i^S
  - if S' follows S …
  - if S' precedes S …
    - Lemma 1 → $w_i^{S'}$ precedes the end of S'

The End - Questions

?