Transaction Systems
Exercise Session 02

Linnea Passing
linnea.passing@tum.de
www-db.in.tum.de

April 25, 2016
Today’s Plan

- (Last week’s homework)
- Review of last week’s lecture
- Page and object model (chapter 2)
- Herbrand semantics, FSR, VSR (chapter 3)
- Homework
Last Week’s Lecture

- Examples: debit/credit, e-commerce, travel planning
- Architectures: two-tier, three-tier
- Concept of transaction: application freed from taking care of concurrency and failures
- Concurrency = overlapping in time
- Parallelism = on multiple processors
- ACID
- Wish list: concurrency control (for isolation), recovery (for atomicity and durability), high throughput and short response times, reliability and availability
- DBMS architecture: language and interface layer, query decomposition and optimization layer, query execution layer, access layer, storage layer
The Page Model

- $D = \{x, y, z, \ldots\}$
- $t_i = p_1 \ldots p_n$, $p_i \in \{r_i(x), w_i(x)\}$ with $x \in D$
- $p_{ij}$ = $j$-th step in transaction $i$
- Partial order: reflexive, antisymmetric, transitive
- Conflict operations: pairs $(r(x), w(x))$ and $(w(x), w(x))$ must be ordered
- Assumptions: each $t$ reads and writes every data item at most once, no read after write, blind writes are possible
The Object Model

- Implicitly includes the page model
- Operations on arbitrary kinds of objects
- Transactions are trees, leaves are elementary operations (with same partial order)
- Layered/multilevel $t$: all leaves have the same distance from the root
- Partial order on inner nodes: iff *all* children of one node precede *all* of another
Roadmap

- Introduction is over
- Now: Concurrency control
- Afterwards: Recovery
- Probably not: Coordination of Distributed Transactions
Notions of Correctness for the Page Model

- Concurrency problems: lost-update, inconsistent-read, dirty-read
- History = complete schedule (termination operators for each \( t \))
- \( trans(s) \), \( commit(s) \), \( abort(s) \), \( active(s) \), \( correct(s) \)
- Serializability = schedule is serial or in \( [S] \sim \) of a serial schedule
Herbrand semantics, FSR, VSR

- Credits: Dr. Andrey Gubichev, 2013
Herbrand semantics: idea

- Read operation $r_i(x)$ reads the value written by the last $w_j(x)$ before $r_i(x)$
- Write operation $w_i(x)$ writes a new value that depends on everything that the transaction $t_i$ has read so far
Herbrand semantics: formalism

- $H_s(r_i(x)) = H(w_j(x))$, $w_j(x)$ is the last write before $r_i$
- $H_s(w_i(x)) = f_{ix}(H_s(r_i(y_1)), \ldots, H_s(r_i(y_m)))$, where $r_i(y_j)$ are all read operations of $t_i$ before $w_i(x)$
- $f_{ix}$ is an uninterpreted $m$-ary function symbol
Herbrand semantics: formalism

- \( H_s(r_i(x)) = H(w_j(x)) \), \( w_j(x) \) is the last write before \( r_i \)
- \( H_s(w_i(x)) = f_{ix}(H_s(r_i(y_1)), \ldots, H_s(r_i(y_m))) \), where \( r_i(y_j) \) are all read operations of \( t_i \) before \( w_i(x) \)
- \( f_{ix} \) is an uninterpreted \( m \)-ary function symbol
- every schedule starts with \( t_0 \): writing all the data items

\[ s = r_1(x)r_2(y) \ldots \]
Herbrand semantics: formalism

- $H_s(r_i(x)) = H(w_j(x))$, $w_j(x)$ is the last write before $r_i$
- $H_s(w_i(x)) = f_{ix}(H_s(r_i(y_1)), \ldots, H_s(r_i(y_m)))$, where $r_i(y_j)$ are all read operations of $t_i$ before $w_i(x)$
- $f_{ix}$ is an uninterpreted $m$-ary function symbol
- every schedule starts with $t_0$: writing all the data items

$$s = w_0(x)w_0(y)c_0r_1(x)r_2(y)\ldots$$
Herbrand semantics: example

\[ s = w_0(x)w_0(y)c_0r_1(x)r_2(y)w_2(x)w_1(y)c_2c_1 \]

- \( H_s(w_0(x)) = f_{0x}() \)
- \( H_s(w_0(y)) = f_{0y}() \)
Herbrand semantics: example

\[ s = w_0(x) w_0(y) c_0 r_1(x) r_2(y) w_2(x) w_1(y) c_2 c_1 \]

- \( H_s(w_0(x)) = f_{0x}() \)
- \( H_s(w_0(y)) = f_{0y}() \)
- \( H_s(r_1(x)) = H_s(w_0(x)) = f_{0x}() \)
- \( H_s(r_2(y)) = H_s(w_0(y)) = f_{0y}() \)
Herbrand semantics: example

\[ s = w_0(x)w_0(y)c_0r_1(x)r_2(y)w_2(x)w_1(y)c_2c_1 \]

- \[ H_s(w_0(x)) = f_0x() \]
- \[ H_s(w_0(y)) = f_0y() \]
- \[ H_s(r_1(x)) = H_s(w_0(x)) = f_0x() \]
- \[ H_s(r_2(y)) = H_s(w_0(y)) = f_0y() \]
- \[ H_s(w_2(x)) = f_2x(H_s(r_2(y))) = f_2x(f_0y()) \]
- \[ H_s(w_1(y)) = f_1y(H_s(r_1(x))) = f_1y(f_0x()) \]
Herbrand semantics: schedule

\[ H[s](x) = H(w_j(x)), \quad w_j(x) \text{ is the last operation writing } x, \text{ for each } x \]
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2w_1(y)c_1 \]
Final State Serializability

Two schedules $s$ and $s'$ are final state equivalent, iff $\text{op}(s) = \text{op}(s')$ and $H[s] = H[s']$
Example

\[ s = w_0(x)w_0(y)c_0 r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2 w_1(y)c_1 \]

equivalent to

\[ t_2 t_1 \]
One more extension to the schedule

\[ s = w_0(x)w_0(y)c_0 r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2 w_1(y)c_1 r_\infty(x)r_\infty(y)c_\infty \]
The *reads-from* relation of $s$:

$$RF(s) = \{(t_i, x, t_j) \mid r_j(x) \text{ reads } x \text{ from } w_i(x)\}$$
Alive steps

- Step $p$ is directly useful for step $q$ ($p \rightarrow q$), if
  - $q$ reads from $p$
  - or $p$ is read and $q$ is subsequent write in the same transaction
- Transitive closure of $\rightarrow$: $\longrightarrow$
- A step $p$ is alive if it is useful for some step from $t_\infty$
Example

\[ s = r_1(x)r_2(y)w_1(y)w_2(y)c_1c_2r_\infty(x)r_\infty(y)c_\infty \]
Example

\[ s = r_1(x)r_2(y)w_1(y)w_2(y)c_1 c_2 r_{\infty}(x)r_{\infty}(y)c_{\infty} \]

- \( w_2(y) \) is alive
- \( r_2(y) \) is alive
- \( r_1(x) \) is not alive
Live Reads-from

The *live reads-from* relation of $s$:

$$LRF(s) = \{(t_i, x, t_j) \mid \text{an alive } r_j(x) \text{ reads } x \text{ from } w_i(x)\}$$
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)r_2(y)w_1(y)w_2(y)c_1c_2r_{\infty}(x)r_{\infty}(y)c_{\infty} \]
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)r_2(y)w_1(y)w_2(y)c_1c_2r_\infty(x)r_\infty(y)c_\infty \]

- \( RF(s) = \{(t_0, x, t_1), (t_0, y, t_2), (t_0, x, t_\infty), (t_2, y, t_\infty)\} \)
- \( LRF(s) = \{(t_0, y, t_2), (t_0, x, t_\infty), (t_2, y, t_\infty)\} \)
Two schedules are FSR equivalent iff

\[ \text{op}(s) = \text{op}(s') \]
\[ LRF(s) = LRF(s') \]
Two schedules are *view equivalent*, iff

- $op(s) = op(s')$
- $H[s] = H[s']$
- $H_s(p) = H_{s'}(p)$ for every read or write step $p$
Two schedules are *view equivalent*, iff

- \( \text{op}(s) = \text{op}(s') \)
- \( \text{RF}(s) = \text{RF}(s') \)
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2w_1(y)c_1 \]
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2w_1(y)c_1 \]

- \( RF(s) = \{(t_0, x, t_1), (t_1, x, t_2), (t_0, y, t_2), (t_1, x, t_\infty), (t_1, y, t_\infty)\} \)
- \( LRF(s) = \{(t_0, x, t_1), (t_1, x, t_\infty), (t_1, y, t_\infty)\} \)
Example

\[ s = w_0(x)w_0(y)c_0r_1(x)w_1(x)r_2(x)r_2(y)w_2(y)c_2w_1(y)c_1 \]

\[ RF(s) = \{(t_0, x, t_1), (t_1, x, t_2), (t_0, y, t_2), (t_1, x, t_\infty), (t_1, y, t_\infty)\} \]

\[ LRF(s) = \{(t_0, x, t_1), (t_1, x, t_\infty), (t_1, y, t_\infty)\} \]

\[ \text{not in VSR (but in FSR)} \]
Herbrand semantics, FSR, VSR

- Credits: Dr. Andrey Gubichev, 2013
Homework

- Already uploaded to our website.