Wait Depth Limited Concurrency Control

Steven Xu
January 31, 2011

Introduction

- Processing power has improved more significantly than data access times.
- Higher data contention follows from the greater ability for concurrency.
- Algorithms must be developed to increase transaction throughput.

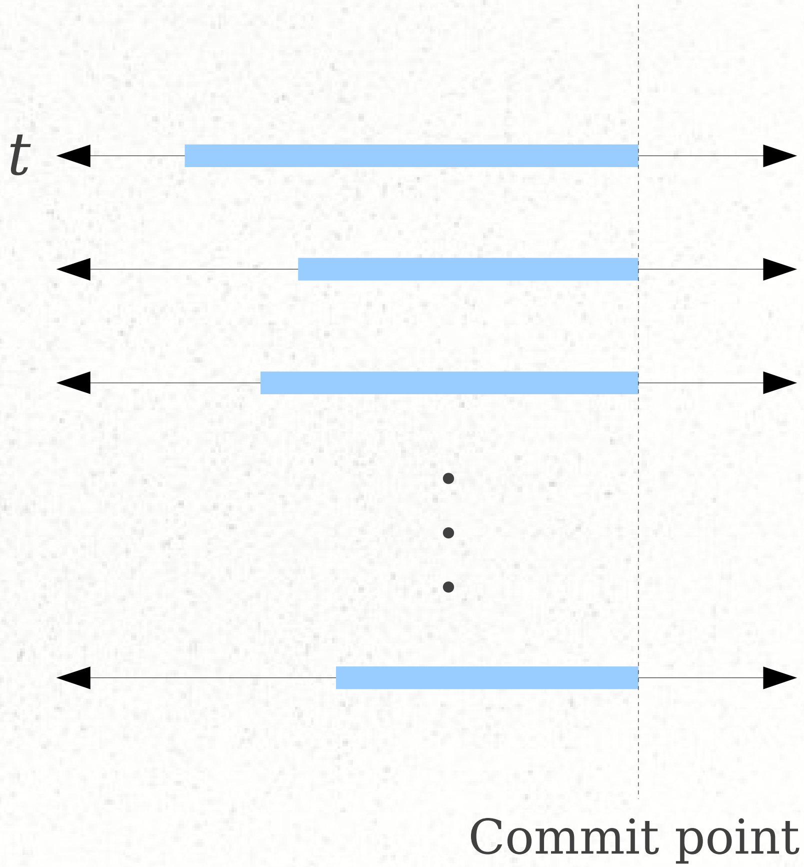
Previous Solutions

- Optimistic restart methods
    - No locking is used.
    - After performing a transaction, the retrieved information is then verified.
    - If the retrieved information is not valid, the transaction could be restarted.
    - If it is valid, then all other transactions sharing the same resources could be restarted.

[Franaszek at al., 1991]

Quadratic effect

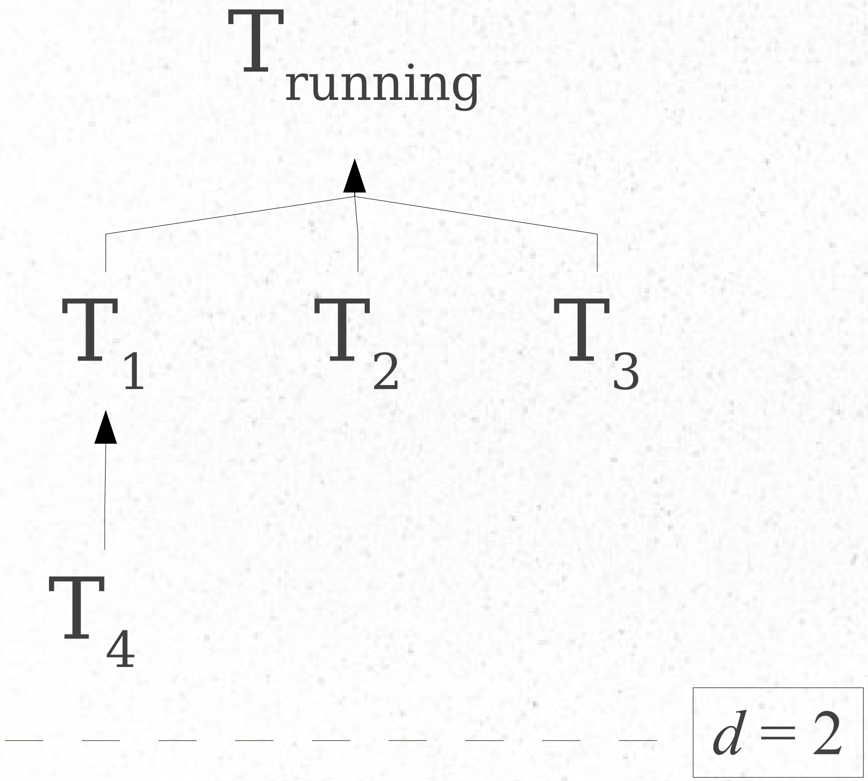
- Transactions that access more resources often take longer.
- This increases the chance that it is restarted.



Paper

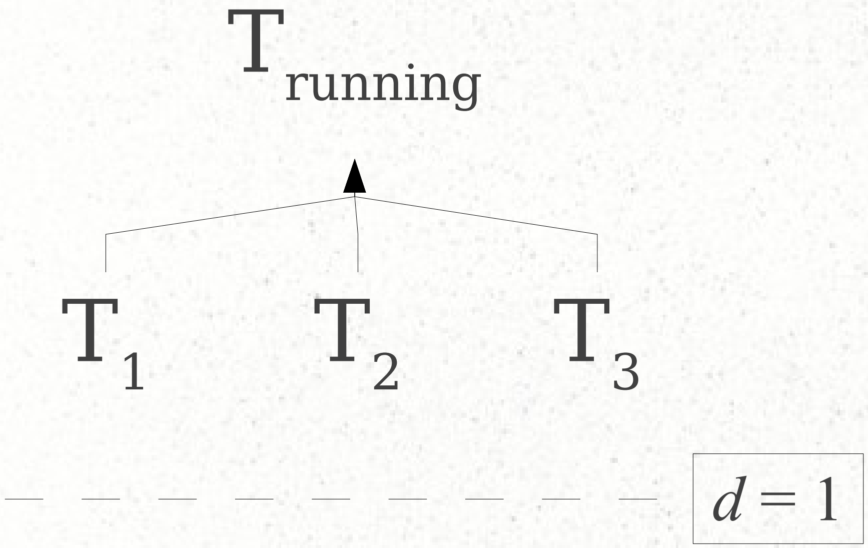
- P. A. Franaszek, J. T. Robinson, and A. Thomasian. Wait Depth Limited Concurrency Control. Proceedings of Seventh International Conference on Data Engineering, 92-101, 1991.
- Part of the IBM Research Division

WDL(*d*)



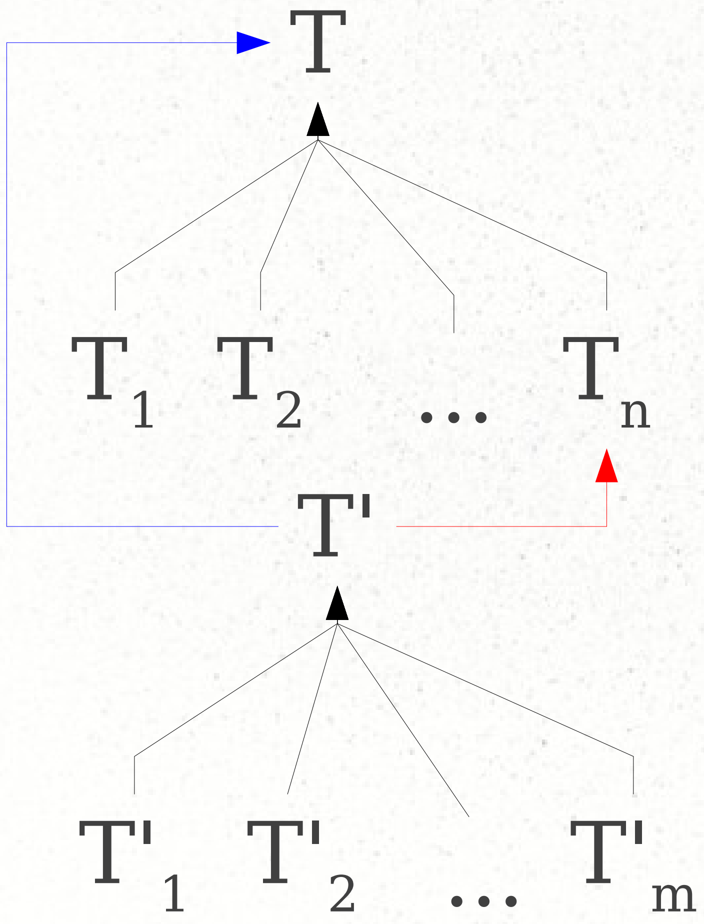
- WDL(*d*) is a class of methods that limit the depth of the tree waiting on a transaction to a depth of *d*.

WDL(1)



- WDL(1) is mainly of interest:
  - Transactions can only be waiting on other running transactions
  - Avoids deadlock

Particular Method of WDL(1)



← T' waits on T.

← If $L(T') \geq L(T)$ and for all i, $L(T') \geq L(T'_i)$, restart T. Otherwise, restart T'.

← If $L(T_n) \geq L(T)$ and $L(T_n) \geq L(T')$, restart T. Otherwise, restart T'.

← If $L(T') \geq L(T_n)$ and for all i, $L(T') \geq L(T'_i)$, restart T_n. Otherwise, restart T'.

Future

Tasks

- Implement WDL(1)
- Implement other concurrency control methods
- Compare their performance under different situations

Challenges

- Associating abstract concepts with their concrete counterparts
- Develop system of common resources fit for all algorithms