dOPT

- Proposed by C. A. Ellis in 1989
- dOPT makes locking unnecessary, Ellis claims
  - Others dispute this, but this is covered later
- dOPT allows a user’s changes to be seen immediately while maintaining consistency
- dOPT is fully distributed and resilient to site failure
How it works

- Each machine has an identifier $i$, which could be an IP or MAC.
- Each machine also maintains a state vector $s$ with $n$ components where $n$ is the number of machines.
- The $i$ component stores the number of changes recorded by the $i$th machine.
Requests

- Requests are of the form: `<i, s, o, p>`
- `i` - the identifier
- `s` - the state vector of the sending machine
- `o` - the operation to be performed
- `p` - the priority, ignore this because it is wrong
Handling Requests

- Action is based on comparison between $s_i$ and $s_x$ the states of the sending and receiving machines.
  - If $s_i = s_x$: execute $o$ immediately
  - If $s_i > s_x$: queue $o$ and execute later
  - If $s_i < s_x$: transform $o$ and execute
Why It Doesn’t Work

- The priority suggested doesn’t work for two clients, suggest a different model for priority
- This suggested model require point to point communication, but this can be expanded using a tree.
- Implies a client-server model
- Load-Balancing also seems possible using this model.
Locking

- Ellis claimed locking was unnecessary, but others argue otherwise.
- Chengzheng Sun argues that locking and operational transforms solve different problems.
An Example

- Consider the following text with a grammar error:
  - “The dog eat dog food”

- Two users noticed this mistake and correct it simultaneously.
  - One changes it to: “The dog eats dog food”
  - The other changes it to: “The dog does eat dog food”
An Example, cont’d.

- dOPT would then consolidate these changes to:
  - “The dog *does* eat dog food”
- This is not what either user intended, nor is it proper English.
Conclusion

- Locking would have prevented both users from changing this phrase simultaneously.
- These sort of problems would be even harder to detect in code than in English, so we need to consider this...
Sources

- C.A. Ellis: Concurrency Control in Groupware Systems
  - http://portal.acm.org/citation.cfm?id=66963&coll=GUIDE&dl=ACM&CFID=56244673&CFTOKEN=48388425

- Gordon V. Cormack: A Counterexample to the Distributed Operational Transform and a Corrected Algorithm for Point-to-point Communication

- Chengzheng Sun: Optional Locking Integrated with Operational Transformation in Distributed Real-Time-Group Editors
  - http://portal.acm.org/citation.cfm?id=301322&coll=portal&dl=ACM&CFID=56244468&CFTOKEN=72761822&ret=1#Fulltext