Concurrency Control

- Locking Algorithms (2PL)

  Shared Read Locks

  Exclusive Write Locks

  Inplace Writes at Access Time

- Optimistic Algorithms (OPT)

  Unrestricted Reads

  Private Writes

  Inplace Writes at Commit Time

Locking versus Optimistic CC

	2PL	OPT
Conflict Detection	Immediate	Validation
Conflict Resolution	Blocking	Restarts

2PL – Conserves resources
OPT – Wastes resources

2PL beats OPT in conventional DBMS
(Carey's Law)

2PL-HP (High Priority)

if Priority (Requester) > Priority (Holder)

restart Holder
grant lock to Requester

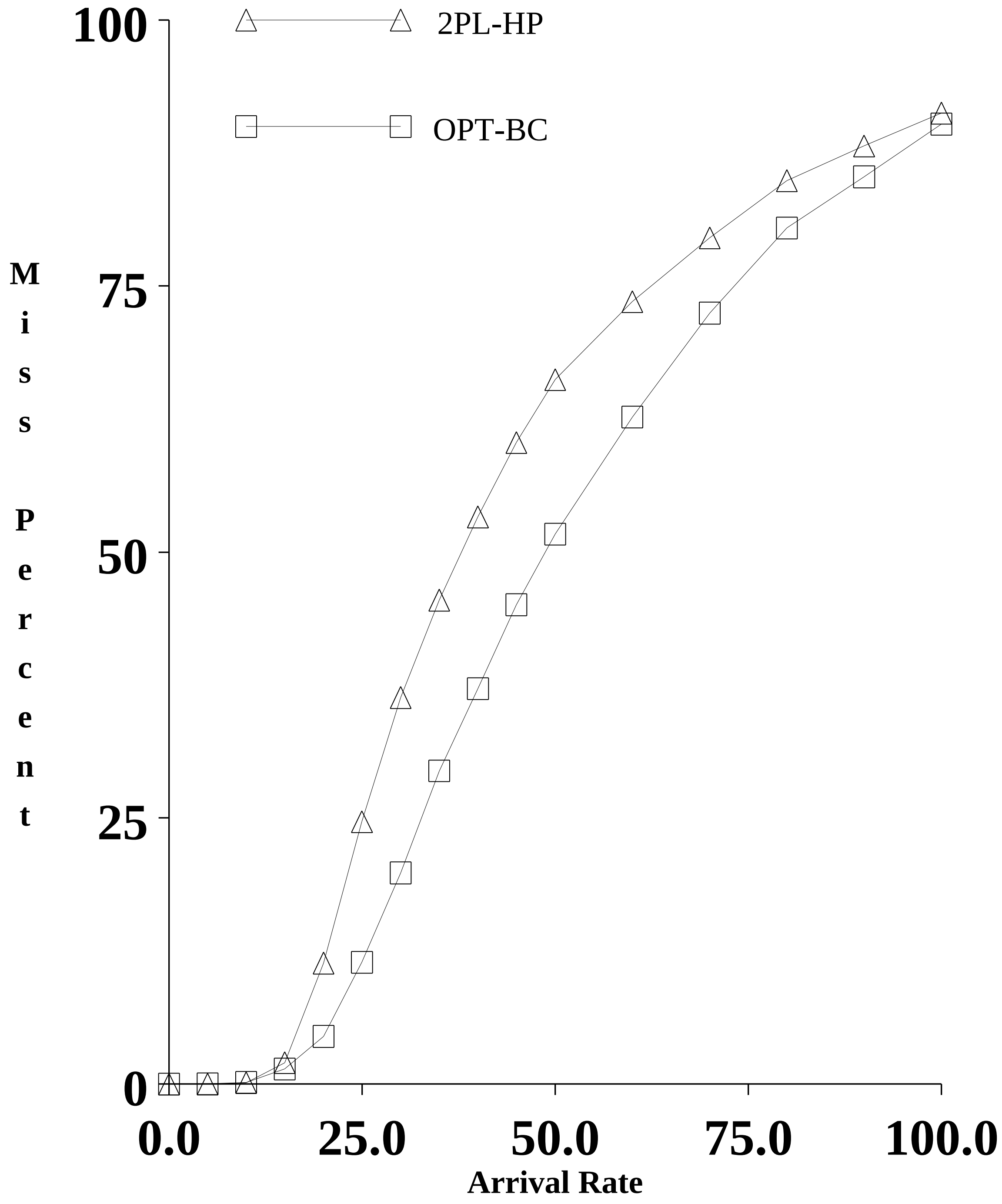
else

block Requester
wait for lock release

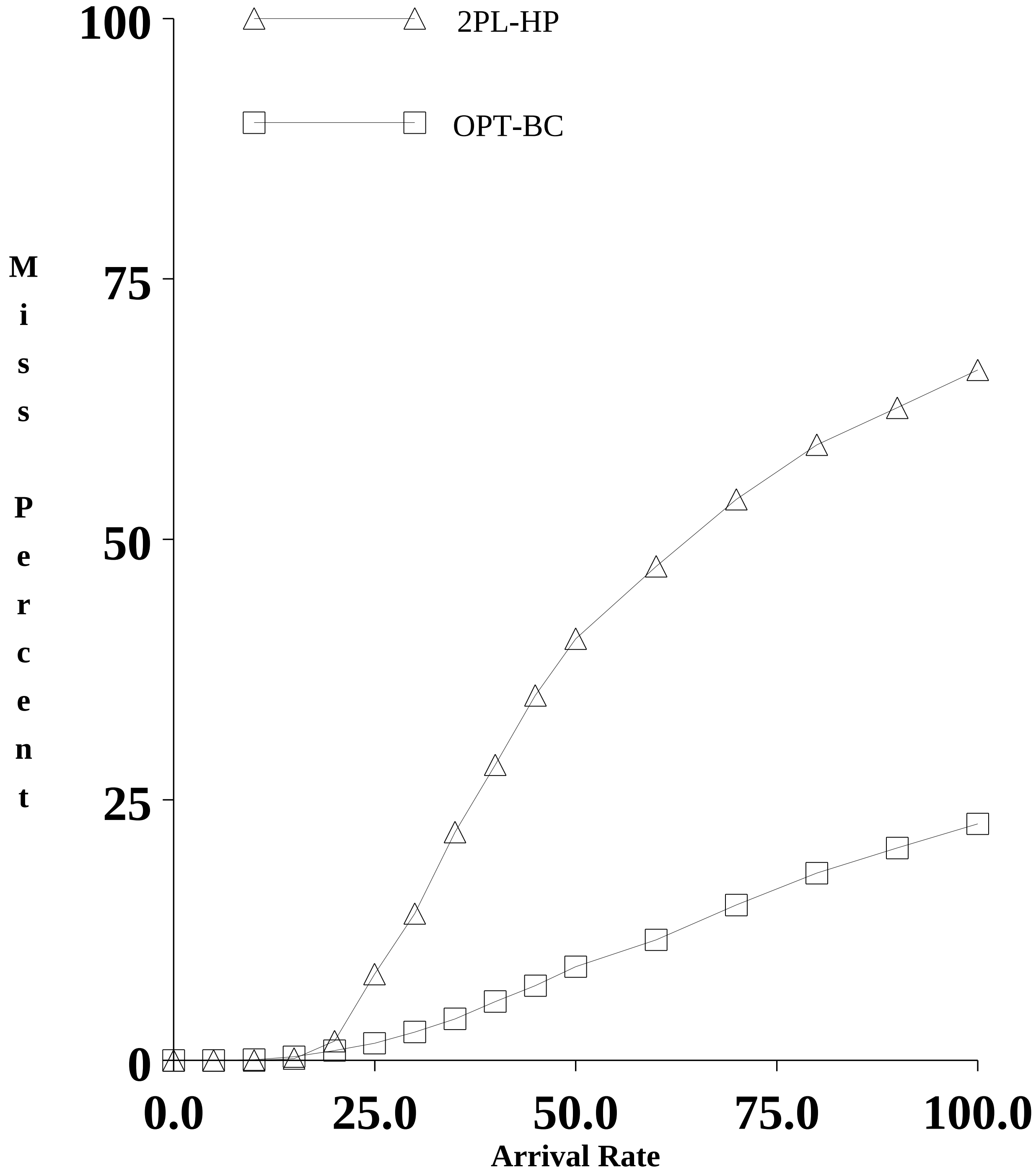
OPT-BC (Broadcast Commit)

restart all conflicting transactions
commit the validating transaction

Resource and Data Contention

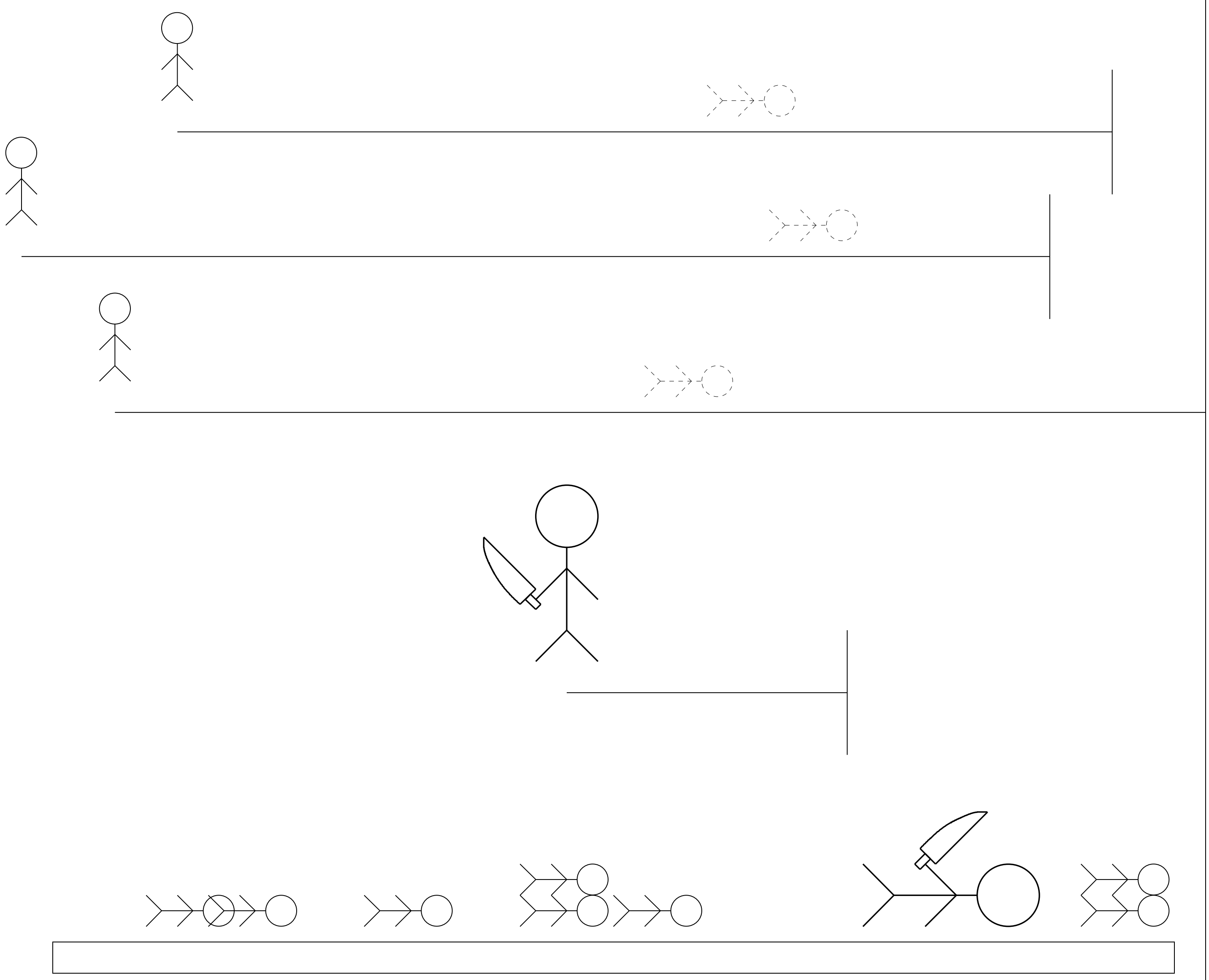


Pure Data Contention



Why Optimistic CC in RTDBS ?

Because transactions may be discarded !



Study 2: Prioritized OPT Algorithms

- Priority-Restart

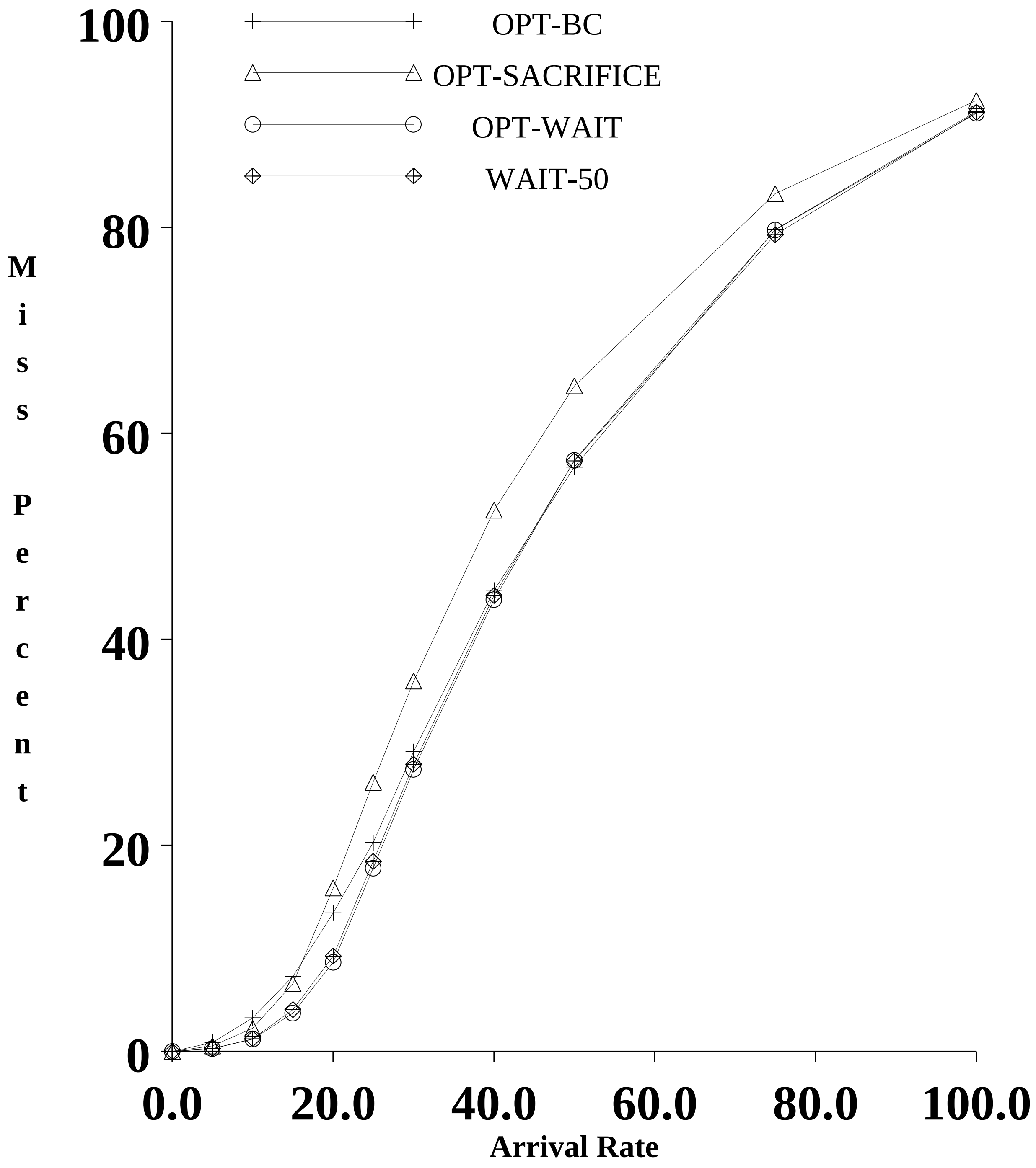
  OPT-SACRIFICE
  restart if conflict with
  higher priority transaction

- Priority-Wait

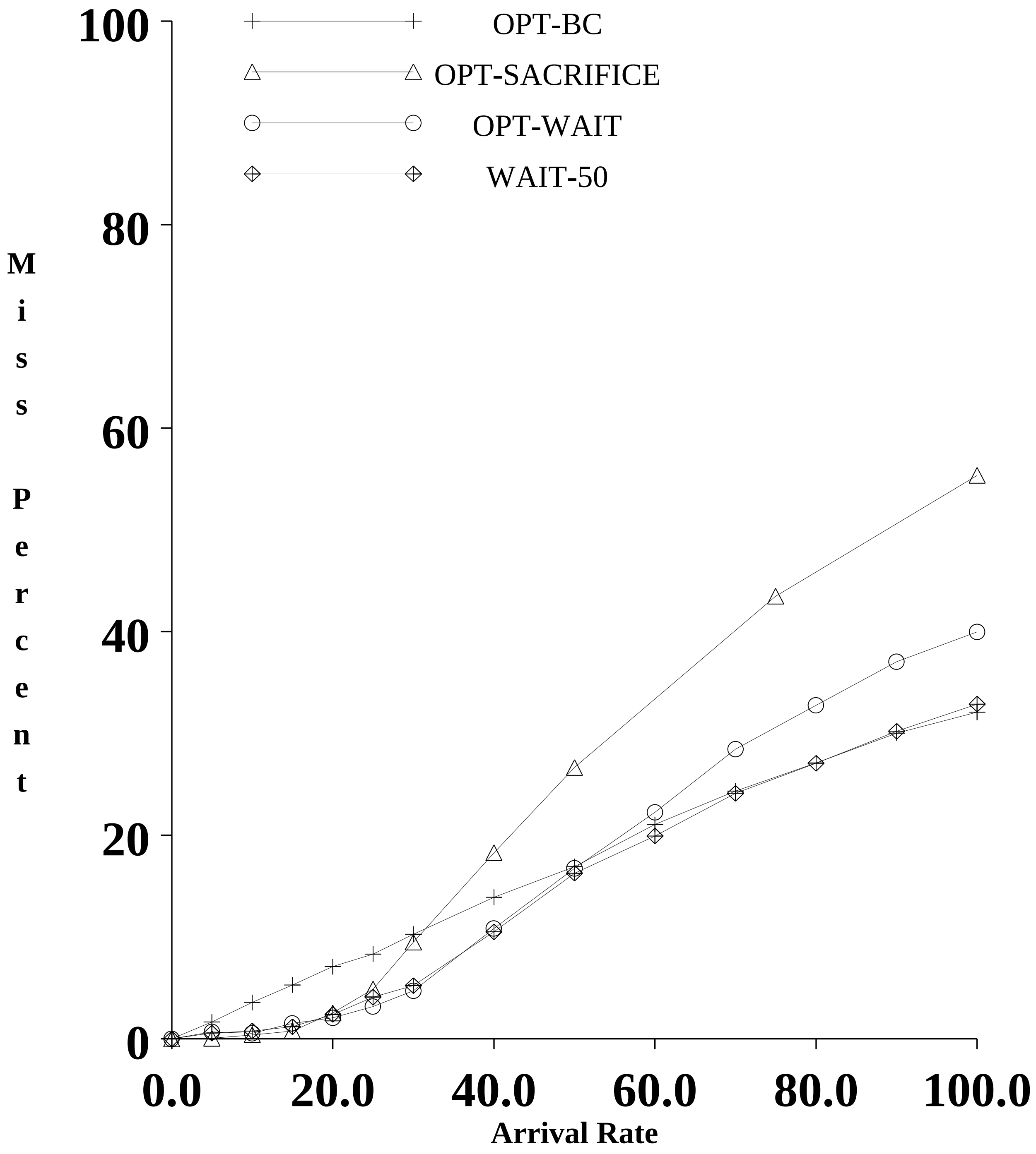
  OPT-WAIT
  wait if conflict with
  higher priority transaction

  WAIT-50
  wait if high priority component
  of conflict set $\geq$ 50 percent

Resource and Data Contention



Pure Data Contention



Conclusions:

- Optimistic is preferable to Locking for firm-deadline RTDBS
- Priority-waiting is the best approach to prioritizing optimistic algorithms
- WAIT-50 detects when the beneficial aspects of waiting are outweighed by its drawbacks

Moral:

Haste makes waste !

Dovernyai v provernai ...
(Trust but verify !)