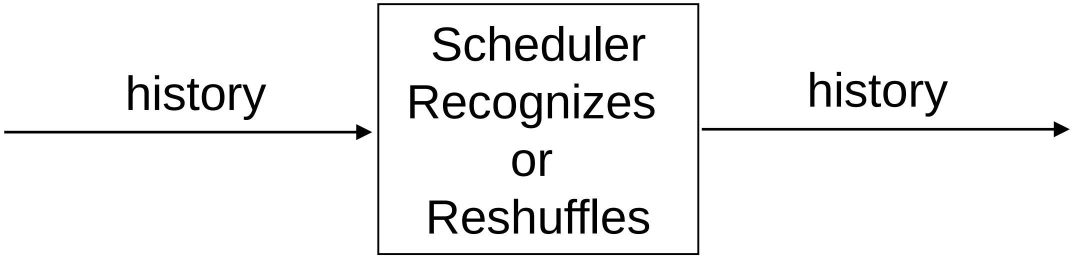
Evaluation Criterion

1. Degree of Concurrency



Less reshuffle ⇒ High degree of concurrency

2. Resources used to recognize
   - Lock tables
   - Time stamps
   - Read/write sets
   - Complexity
3. Costs
   - Programming ease

General Comments

- Information needed by Concurrency Controllers
  - Locks on database objects (System-R, Ingres, Rosenkrantz...)
  - Time stamps on database objects (Thomsa, Reed)
  - Time stamps on transactions (Kung, SDD-1, Schlageter, Bhargava...)

- Observations
  - Time stamps mechanisms more fundamental than locking
  - Time stamps carry more information
  - Checking locks costs less than checking time stamps

General Comments (cont.)

- When to synchronize
  - First access to an object (Locking, pessimistic validation)
  - At each access (question of granularity)
  - After all accesses and before commitment (optimistic validation)
- Fundamental notions
  - Rollback
  - Identification of useless transactions
  - Delaying commit point
  - Semantics of transactions

Probability that two transactions do not share an object

$$= \frac{{}^{M}C_{B_S} * {}^{M-B_S}C_{B_S}}{{}^{M}C_{B_S} * {}^{M}C_{B_S}}$$

$$= \left(\frac{M - B_S}{M}\right) * \left(\frac{M - B_S - 1}{M - 1}\right) * \left(\frac{M - 2B_S + 1}{M - B_S + 1}\right)$$

Lower bound on this problem $= \left(\frac{M - 2B_S + 1}{M - B_S + 1}\right)^{B_S}$

Maximum problem that two transactions will share an object

$$= 1 - \left(\frac{M - 2B_S + 1}{M - B_S + 1}\right)^{B_S}$$

BS	M	Probability of conflict
5	100	.0576
10	500	.0025
20	1000	.113

Probability of cycle
= 0(PC^2)
≅ small

Concurrency/Multiprogramming level is low

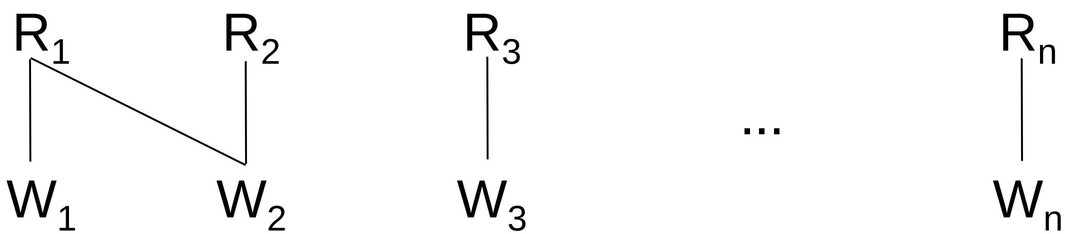
Example:

I/O	=	.005 seconds
CPU	=	.0001 seconds
Trans size	=	5
Time to execute trans.	=	.0255 seconds

For another trans. to meet this trans. in the system

$$\text{Arrival rate} > \frac{1}{.0255} \text{ or} > 40 \text{ per second}$$

Example: $h = R_1 R_2 W_2 R_3 W_3 \ldots R_n W_n W_1$



Locking: This history not allowed

W_2 is blocked by R_1

T_2 cannot finish before T_1

What if T_1 is a log trans. and T_2 is a small trans.?

T_1 blocks T_2; can block T_3 … T_n if $\left(R_2 \cap W_2 \neq \phi\right)$

Optimistic [Kung]

T_i ($i = 2,\ldots,n$) commit. W_i saved for $valid_n$

R_1 validated with W_i, T_1 aborted

$$h = R_1 R_2 W_2 \ldots R_n W_n W_1$$

switch to

Optimistic Validation (first modification)

$$h = R_1 R_2 W_2 R_3 W_3 \ldots R_n W_n W_1$$

Try this or this switch

T_i's can commit, W_i and R_i saved from validation

W_1 validates with W_i and R_i

T_1 aborted if validation fails (second modification)

$$h = R_1 R_2 W_2 R_3 W_3 \ldots R_n W_n W_1$$

Switch R_1 to the right after W_2, $W_3 \ldots W_n$

Switch W_1 to the left before R_n, $R_{n-1} \ldots R_2$

If R_1 and W_1 are adjacent, T_1 is successful

$$h \equiv R_1 R_2 W_2 \ldots R_k W_k \ldots R_n W_n W_1$$

$$\equiv R_2 W_2 \ldots R_1 W_1 R_k W_k \ldots R_n W_n$$