Problem Solving in Computer Science
Scribe notes by Salman Mirghasemi
06.11.2008

1 BDD Operations

Suppose that $b_0$ and $b_1$ are the two BDDs shown in figure 1.

![Two binary decision diagrams $b_0$ and $b_1$](image)

Figure 1: Two binary decision diagrams $b_0$ and $b_1$

$$b_0 = (\neg x_i \land b_{00}) \lor (x_i \land b_{01})$$
$$b_1 = (\neg x_j \land b_{10}) \lor (x_j \land b_{11})$$

1. Conjunction

- $i < j : b_0 \land b_1 = (\neg x_i \land (b_{00} \land b_{11})) \lor (x_i \land (b_{01} \land b_{11}))$
- $i = j : b_0 \land b_1 = (\neg x_i \land (b_{00} \land b_{10})) \lor (x_i \land (b_{01} \land b_{11}))$

2. Disjunction

- $i < j : b_0 \lor b_1 = (\neg x_i \land (b_{00} \lor b_{11})) \lor (x_i \land (b_{01} \lor b_{11}))$
- $i = j : b_0 \lor b_1 = (\neg x_i \land (b_{00} \lor b_{10})) \lor (x_i \land (b_{01} \lor b_{11}))$

3. Negation

$$\neg b_0 = (\neg x_i \land \neg b_{00}) \lor (x_i \land \neg b_{01})$$

2 Concurrnet BDD computation

The BDD pool is in the shared memory and concurrent threads make a stream of MakeBDD calls to that shared structure.
In concurrent operations two properties should be considered: safety and liveness.

1. Safety
   No call to MakeBDD returns a "wrong" answer.
   
   Linearizability: There is a sequential interleaving of all makeBDD calls such that the return values of all concurrent calls are exactly those that would be obtained from sequentially executing all calls in the order of the interleavings.

2. Liveness
   Wait Freedom: Each method call return within a finite number of steps.
   
   Lock Freedom: There is always some method call that returns in linear number of steps (guarantees overall progress but not progress on each thread).

A good synchronization solution locks as little data and as short time is possible.
You can find examples for Course-Grained and Fine-Grained synchronization in this book: "The Art of Multiprocessor Programming, Written by H.Maurice and N.Shavit".