Prof. Bernd Finkbeiner
Dr. Calogero Zarba
Moritz Hahn

Midterm Solutions

Embedded Systems

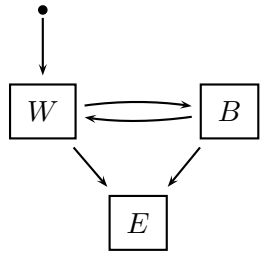
Problem 1 (Embedded Systems)

(a) Embedded systems are computer systems that are encapsulated into larger products, and that are normally not directly visible to the user. Give a concise definition of the term "embedded system."

(b) Car, train, airplane, mobile phone, TV, microwave, fridge, pedometer, heart monitor, stop watch.

(c) Dependable (reliable, manteinable, available, safe, secure), efficient (enery, code size, run time, weight cost), specific, real-time, hybrid, reactive.

Problem 2 (Statecharts)

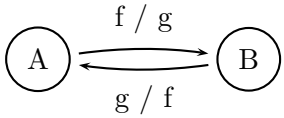


Initial assignment: `poolW := 300, poolB:= 300, grace := 30`

Transitions:

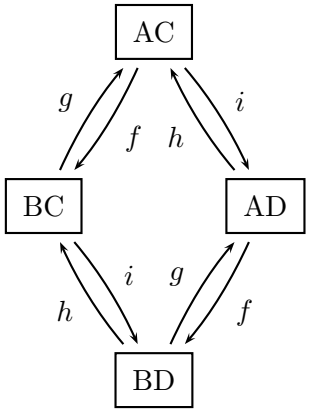
- $W \to B$: `white_moves / poolW := poolW + 12; grace := 30`
- $B \to W$: `black_moves / poolB := poolB + 12; grace := 30`
- $W \to W$: `tick[grace > 0] / grace := grace - 1`
- $W \to W$: `tick[grace = 0 and poolW > 0] / poolW := poolW - 1`
- $W \to E$: `tick[poolW = 0] / black_wins`
- $B \to B$: `tick[grace > 0] / grace := grace - 1`
- $B \to B$: `tick[grace = 0 and poolB > 0] / poolB := poolB - 1`
- $B \to E$: `tick[poolB = 0] / white_wins`

Problem 3 (Statecharts)

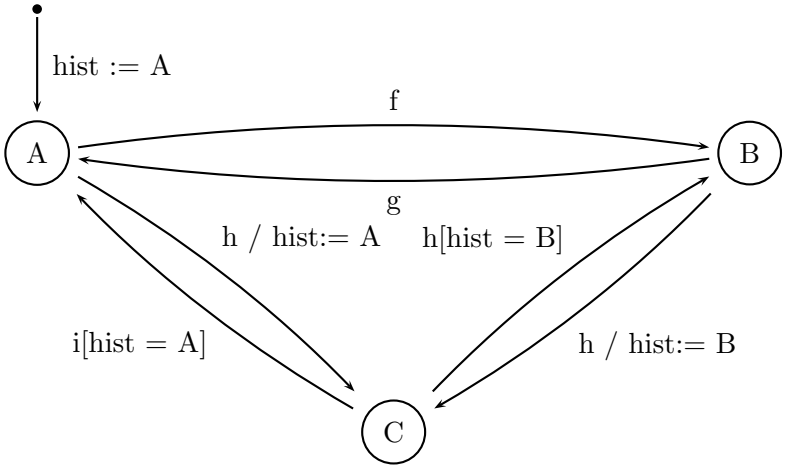


Problem 4 (Statecharts)

Draw a statechart equivalent to the one shown below, but that uses no concurrency. For simplicity, assume that at most one event can occur at each unit of time.



Problem 5 (Statecharts)



Problem 6 (Petri nets)

- a_i, b_i: exactly one player has the right to move at table i, for $i = 1, 2, 3$;
- $\{c_1, c_2, c_3\}$: the Grandmaster is always in front of exactly one table.

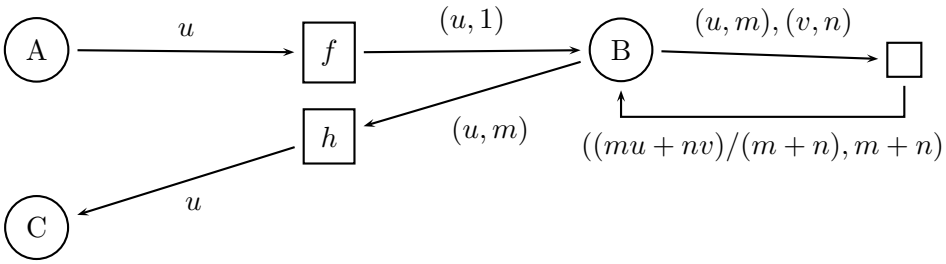
Problem 7 (Petri nets)

(a) $max(S)$;

(b) replace the edgel labelled with $max(u, v)$ with an edge labelled with $\min(u, v)$;

(c) replace the edgel labelled with $max(u, v)$ with an edge labelled with $u + v$;

(d) The transition h has lower priority than g:



Problem 8 (Kahn process networks)

In the following, the following C-like code is used:

- Process 1

```
send(1, out);
while (true) {
  wait(a, in);
  send(a, out);
}
```

- Process s

```
while (true) {
  wait(a, in);
  send(a+1, out);
}
```

- Process D

```
while (true) {
  wait(a, in);
  send(a, out1);
  send(a, out2);
}
```

- Process $\boxed{+}$

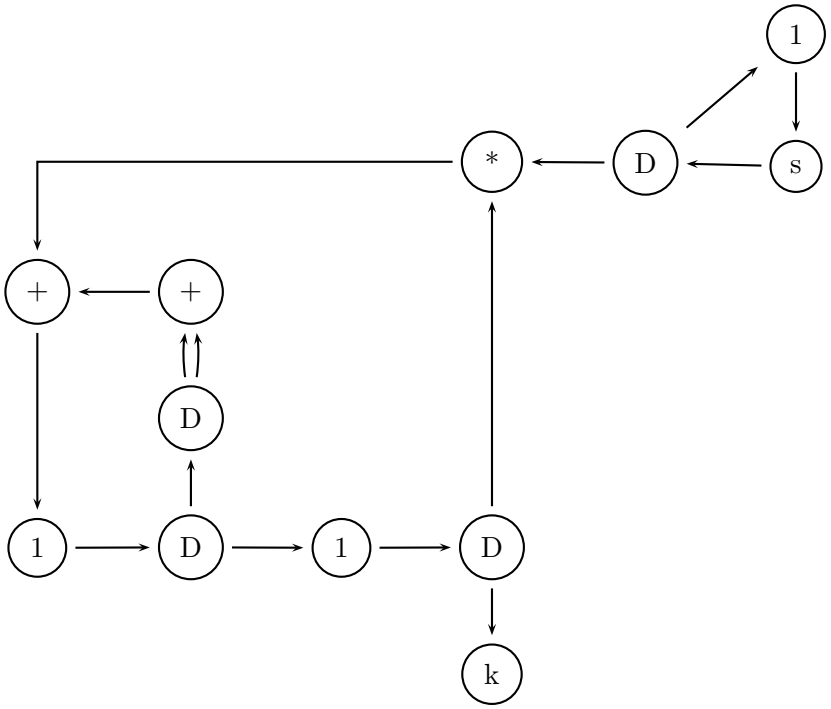
```
while (true) {
  wait(a, in1);
  wait(b, in2);
  send(a+b, out);
}
```

- Process $\boxed{*}$

```
while (true) {
  wait(a, in1);
  wait(b, in2);
  send(a*b, out);
}
```

- Process $\boxed{\text{k}}$

```
while (true) {
  wait(a, in);
}
```



Problem 9 (SDF networks)

The net is schedulable by $(ABB)^*$.