COMP30112: Concurrency

Topics 5.3: Liveness Properties

Howard Barringer

Room KB2.20: email: Howard.Barringer@manchester.ac.uk

April 2009
Outline

Topic 5.3: Liveness Properties

- Progress Properties
- Priority
- Example - Single Lane Bridge again
- Java Implementation for Fair Bridge
Outline

Topic 5.3: Liveness Properties

Progress Properties

Priority

Example - Single Lane Bridge again

Java Implementation for Fair Bridge
Making Progress

- **Liveness**: something good *eventually* happens
- **Progress**: an action always *eventually* gets executed
- **Fair Choice**: if a choice over a set of transitions is made infinitely often, then every transition in the set will be chosen infinitely often

FSP uses *Fair Choice* by default

\[
\text{progress } P = \{a_1, \ldots, a_n\}
\]

defines Progress Property \( P \):

*At least one of the actions \( a_1, \ldots, a_n \) will be executed infinitely often*
Example: Coins again

\[
\text{TWOCOIN} = ( \text{pick} \rightarrow \text{COIN} \\
\quad | \text{pick} \rightarrow \text{TRICK} ),
\]

\[
\text{TRICK} = ( \text{toss} \rightarrow \text{heads} \rightarrow \text{TRICK} ),
\]

\[
\text{COIN} = ( \text{toss} \rightarrow \text{heads} \rightarrow \text{COIN} \\
\quad | \text{toss} \rightarrow \text{tails} \rightarrow \text{COIN}).
\]

\[
\text{progress HEADS} = \{\text{heads}\} \\
\text{progress TAILS} = \{\text{tails}\} \\
\text{progress HEADSorTAILS} = \{\text{heads, tails}\}
\]
LTS for TWOCOIN
Outline

**Topic 5.3: Liveness Properties**
- Progress Properties
- Priority
  - Example - Single Lane Bridge again
  - Java Implementation for Fair Bridge
Terminal Sets

A terminal set of states is one where every state is reachable from every other, and no transition to any state outside terminal set.

i.e. terminal set = strongly connected component

A progress property is violated if there is a terminal set in which none of the progress set actions appear.

Default progress: each action as a separate progress property
Adding Priority to Actions

**High Priority:** $P <\{a_1, \ldots, a_n\}$

If they occur in a choice, do $a_1, \ldots, a_n$ in preference to other actions.

**Low Priority:** $P >\{a_1, \ldots, a_n\}$

If they occur in a choice, do other actions in preference to $a_1, \ldots, a_n$

i.e. discard lower priority actions within a choice.

*Utilise to uncover potential progress problems.*
Outline

Topic 5.3: Liveness Properties
  Progress Properties
  Priority
  Example - Single Lane Bridge again
  Java Implementation for Fair Bridge
Congestion on the Single-Lane Bridge

progress BLUECROSS = { blue[ID].enter }
progress REDCROSS = { red[ID].enter }

||CongestedBridge = SingleLaneBridge
    >> { red[ID].exit, blue[ID].exit }.

★ What is result? ★    ★ How can we fix it? ★
First Attempt

\[
\text{CAR} = (\text{request} \rightarrow \text{enter} \rightarrow \text{exit} \rightarrow \text{CAR}).
\]

\[
\text{BRIDGE} = \text{BRIDGE}[0][0][0][0],
\text{BRIDGE}[\text{nr}:T][\text{nb}:T][\text{wr}:T][\text{wb}:T] =
\begin{align*}
\text{(red[ID].request} & \rightarrow \text{BRIDGE}[\text{nr}][\text{nb}][\text{wr}+1][\text{wb}] \\
\mid \text{when (nb==0 && wb==0)} & \\
\text{red[ID].enter} & \rightarrow \text{BRIDGE}[\text{nr}+1][\text{nb}][\text{wr}][\text{wb}] \\
\text{red[ID].exit} & \rightarrow \text{BRIDGE}[\text{nr}][\text{nb}][\text{wr}][\text{wb}] \\
\text{blue[ID].request} & \rightarrow \text{BRIDGE}[\text{nr}][\text{nb}][\text{wr}][\text{wb}+1] \\
\mid \text{when (nr==0 && wr==0)} & \\
\text{blue[ID].enter} & \rightarrow \text{BRIDGE}[\text{nr}][\text{nb}+1][\text{wr}][\text{wb}][\text{wb}] \\
\text{blue[ID].exit} & \rightarrow \text{BRIDGE}[\text{nr}][\text{nb}][\text{wr}][\text{wb}]
\end{align*}
\]

★ Why does this not work? ★
BRIDGE = BRIDGE[0][0][0][0][True],
BRIDGE[nr:T][nb:T][wr:T][wb:T][bt:B] =
    (red[ID].request -> BRIDGE[nr][nb][wr+1][wb][bt]
     |when (nb==0 && (wb==0 || !bt))
     red[ID].enter -> BRIDGE[nr+1][nb][wr-1][wb][bt]
     |red[ID].exit -> BRIDGE[nr-1][nb][wr][wb][True]
     |blue[ID].request->BRIDGE[nr][nb][wr][wb+1][bt]
     |when (nr==0 && (wr==0 || bt))
     blue[ID].enter -> BRIDGE[nr][nb+1][wr][wb-1][bt]
     |blue[ID].exit -> BRIDGE[nr][nb-1][wr][wb][False]
).
Topic 5.3: Liveness Properties

Outline

Topic 5.3: Liveness Properties

Progress Properties

Priority

Example - Single Lane Bridge again

Java Implementation for Fair Bridge
Java for a Fair Bridge

class FairBridge extends Bridge {

    private int nred = 0;  private int nblue=0;
    private int waitblue=0; private int waitred=0;
    private boolean blueturn = true;

    synchronized void redEnter()
        throws InterruptedException {
            ++waitred;
            while (nblue>0
                || (waitblue>0 && blueturn)) wait();
            --waitred;
            ++nred; }

    synchronized void redExit(){
        --nred;
        blueturn = true;
        if (nred==0)
            notifyAll(); }

}
synchronized void blueEnter() throws InterruptedException {
    ++waitblue;
    while (nred>0 || (waitred>0 && !blueturn)) wait();
    --waitblue;
    ++nblue;
}

synchronized void blueExit(){
    --nblue;
    blueturn = false;
    if (nblue==0)
        notifyAll();
}
}