I. Introduction

In theoretical computer science a bisimulation is a binary relation between labeled transition systems, associating systems which behave in the same way in the sense that one system simulates the other and vice-versa [4]. Bisimulation find applications in the formal verification of concurrent systems. For example, to check the equivalence of an implementation of a certain system with respect to its specification model. Usually, the process of finding a bisimulation equivalence between two labeled transition systems includes two main steps. First, to minimize each one of them to its canonical form, and, then, perform the comparison between the canonical forms [1]. This project was concerned with the first step, which is, reducing a labeled transition system to a canonical form.

II. Labeled Transition System (LTS) Minimization

A. LTS Definition

A labeled transition system $S$ is defined as $S = (Q, A, T, q_0)$ where $Q$ is a set of states, $A$ is a finite set of actions, $T \subseteq Q \times A \times Q$ is the transition relation and $q_0$ is the initial state.

B. Bisimulation Definition

According to [1], bisimulation could be formally defined as follows: Given a labeled transition system $S = (Q, A, T, q_0)$, a binary relation $\rho \subseteq Q \times Q$ is a bisimulation if and only if:

\[
\forall (p_1, p_2) \in \rho \Rightarrow \forall a \in A.
\]

\[
\forall_{r_1}(p_1 \xrightarrow{a} r_1 \Rightarrow \exists r_2((p_2 \xrightarrow{a} r_2 \land (r_1, r_2) \in \rho) \land \forall_{r_2}(p_2 \xrightarrow{a} r_2 \Rightarrow \exists r_1((p_1 \xrightarrow{a} r_1 \land (r_1, r_2) \in \rho))\}
\]

C. LTS Minimization Algorithm Overview

This task takes an LTS and divides it into its coarsest blocks (i.e., least number of blocks) so that each block consists of a set of bisimulation-equivalent states.
D. Algorithm Example

An example of an input to the LTS Minimization task [1] is shown in Fig. 1(a). The algorithm will divide this LTS into its coarsest blocks as shown in Fig. 1(b), where each block consists of a set of bisimulation-equivalent states. Then, the algorithm should produce the minimized graph as in Fig. 1(c).

![Figure 1(a): An example of an input LTS to the minimization task.](image)

![Figure 1(b): The LTS coarsest blocks.](image)

![Figure 1(c): The minimized LTS.](image)

Figure 1. Example of bisimulation-based minimization of an LTS
III. Some Definitions

A Block is a set of states. For example, in Fig.1, \( B_3 = \{3,4\} \).

A Partition is a set of Blocks. The Blocks that constitute the Partition are mutually exclusive (i.e., don’t have states in common) and their union constitutes the graph universe (i.e., set \( Q \)). For example, in Fig.1, \( P = \{B_1, B_3, B_4\} \), \( Q = \{0,1,2,3,4,5\} \), \( Q = B_1 \cup B_3 \cup B_4 \).

To define graph transitions we use the following terminology: \( T_\alpha[p] = \{q\} \) means an \( \alpha \)-transition from state \( p \) to state \( q \). For example, in Fig.1, \( T_a[0] = \{1\} \) and \( T_b[1] = \{3,4\} \). Similarly, \( T_\alpha^{-1}[q] = p \) means an inverse \( \alpha \)-transition. For example, in Fig.1, \( T_b^{-1}[4] = \{1\} \). We also define an inverse transition for Block \( B \) and action \( \alpha \) as follows: \( T_\alpha^{-1}[B] = \cup\{T_\alpha^{-1}[q] | q \in B\} \). Back to the example in Fig.1, \( T_b^{-1}[B_3] = \{0,1,2\} \).

A set of set called Splitters, \( W \), is used to contain those blocks that are going to be used to split the Partition. \( W \) should become more clear in the next two Sections.

An Info map is defined for a Block \( B \), state \( p \) and action \( a \) as follows:
\[
Info_B(a, p) = |\{\underbrace{T_a[p]}_{\alpha} \cap B| \]
\[
Info_B(b, l) = 2
\]

IV. Pseudo Code of the Algorithm

In this section we introduce a pseudo code of the algorithm we used to do the LTS bisimulation-based minimization. It’s almost the same as that described in [1].

Choose & remove any splitter (\( S \)) in \( W \)

Case \( S \) is a simple splitter composed only of \( B \)

For each \( \alpha \in A \)

Construct \( I = \{X | \exists X \in P \land X_1 = X \cap T_\alpha^{-1}[B] \neq \emptyset\} \)

//i.e., \( I \) is a set of those blocks in \( P \) that are going to be splitted by \( B \)

For each block \( X \) in \( I \):

Compute \( X_1 = X \cap T_\alpha^{-1}[B] \)

Compute \( X_2 = X - T_\alpha^{-1}[B] \)

Compute map info

Check \((X, X_1, X_2)\) & Update \( P \) and \( W \) (if required):

If \((X_1 = \emptyset) \lor (X_2 = \emptyset)\)

Do nothing! //No split

If \((X_1 > \emptyset) \&\& (X_2 > \emptyset)\)

Remove \( X \) from \( P \).
Add $X_1$ to $P$.
Add $X_2$ to $P$.
Add $(X,X_1,X_2)$ to $W$.

Case $S$ is a compound splitter composed of $(B_i,B_{ii},B_{iii})$ // $B = B_i \cup B_{ii}$,
assume $|B_i|<|B_{ii}|$

For each $\alpha \in A$
Construct $I = \{X | X \in P \land X \subseteq T_\alpha^{-1}[B]\}$
//i.e., $I$ is a set of those blocks in $P$ that are going to be splitted by $S$
// NB: $P$ is already refined w.r.t. $B$.
Calculate map info$_B$
For each block $X$ in $I$:
Compute $(X_1,X_2,X_3)$ as follows:
$X_1$: Set of states in $X$ that goes to $B_i$ but not to $B_{ii}$ with action $\alpha$.
$X_2$: Set of states in $X$ that goes to $B_{ii}$ but not to $B_i$ with action $\alpha$.
$X_3$: Set of states in $X$ that goes to both $B_i$ and $B_{ii}$.
// This 3 subblocks are calculated directly from info$_B$ and info$_B_i$ as follows:
For each state $s$ in block $X$:
If (info$_B_i[s][\alpha] ==$ info$_B[s][\alpha]$)
  Add $s$ to $X_1$
  info$_B_{iii}[s][\alpha]=0$
If (info$_B_i[s][\alpha] == 0$)
  Add $s$ to $X_2$
  info$_B_{iii}[s][\alpha]=$ info$_B[s][\alpha]$
  If ((info$_B_i[s][\alpha] > 0$) &&( info$_B_i[s][\alpha]<$
    info$_B[s][\alpha]$ ))
    Add $s$ to $X_3$
    info$_B_{iii}[s][\alpha]=$ info$_B[s][\alpha]-$ info$_B_i[s][\alpha]$

Check $(X_1,X_2,X_3)$ & Update $P$ and $W$ (if required):
If (($X==X_1$) || ($X==X_2$)) || ($X==X_3$))
  Do nothing! //No split
Else
  Replace $X$ in $P$ by non-null $X_1,X_2,X_3$
  Add non-null $X_1,X_2,X_3$ to $W$ in the same way
  as in the simple splitter case, except that if all $X_1,X_2,X_3$ are non-null add the following to $W$: $(X,X_1,X_{23}),(X_{23},X_2,X_3)$
V. C++ Implementation and Minimization Example

The Pseudo code has been implemented in C++. Following is a detailed LTS minimization example:

An input LTS shown in Fig.1(a) is passed to the minimization algorithm. Initially, both Partition, $P$, and Splitters, $W$, will contain the graph universe, $B_0$.

Partition($P$) = \{$B_0$\}; Splitters($W$) = \{\}

$B_0$ = \{0,1,2,3,4,5\}

A = \{a,b,c\}

Choose & remove any splitter ($S$) in $W$ $W=\{}$, working with Splitter $B_0$ 

Case $S$ is a simple splitter composed only of $B$: $B_0$ is a Simple Splitter

For each $\alpha \in A$  Start with action $a$, $T_a^{-1}[B_0]$ = \{0,1,2\}

Construct $I = \{X_1 | \exists X \in P \land X_1 = X \cap T_a^{-1}[B] \neq \phi\}$

//i.e., $I$ is a set of those blocks in $P$ that are going to be splitted by $B$

$I$ = \{\}

For each block $X$ in $I$: $X=B_0$

Compute $X_1 = X \cap T_a^{-1}[B]$ $X_1$ = \{0,1,2\}

Compute $X_2 = X - T_a^{-1}[B]$ $X_2$ = \{3,4,5\}

Compute map info$_B$

info$_B_0$[0][a]=1, info$_B_0$[1][a]=1, info$_B_0$[2][a]=1

info$_B_0$[3][a]=0, info$_B_0$[4][a]=0, info$_B_0$[5][a]=0

Check ($X, X_1, X_2$) & Update $P$ and $W$ (if required):

If ((($X_1=\phi$) || ($X_2=\phi$)) FALSE

Do nothing! //No split

If (((X$>_0$) && ($X_2>_0$)) TRUE

Remove $X$ from $P$.

Add $X_1$ to $P$.

Add $X_2$ to $P$.

Add ($X$, $X_1$, $X_2$) to $W$.

$P$ = \{B_1, B_2\}; $W$ = \{(B_0,B_1,B_2)\}

Where $B_1$ = \{0,1,2\}, $B_2$ = \{3,4,5\}

After refinement with respect to $B_0$ and action a, partition ($P$) will look like the following

$P$ = \{B_1, B_2\}

$W$ = \{(B_0,B_1,B_2)\}

At this point, the LTS Partition, $P$, should look like Fig 2.
Choose & remove any splitter $(S)$ in $W = \{(B_0, B_1, B_2)\}$, Still working with Splitter $B_0$

Case $S$ is a simple splitter composed only of $B$: Still $B_0$ is a Simple Splitter

For each $\alpha \in A$, Next is action $b$, $T_b^{-1}[B_0] = \{0, 1, 2\}$

Construct $I = \{X_1 \mid \exists X \in P \land X = X \cap T_a^{-1}[B] \neq \emptyset\}$

//i.e., $I$ is a set of those blocks in $P$ that are going to be splitted by $B$

$I = \{B_1\}$

For each block $X$ in $I$: $X = B_1$

Compute $X_1 = X \cap T_a^{-1}[B]$ $X_1 = \{0, 1, 2\}$

Compute $X_2 = X - T_a^{-1}[B]$ $X_2 = \{\}$

Compute map info$_B$

info$_B[0][b] = 1$, info$_B[1][b] = 1$, info$_B[2][b] = 2$, info$_B[3][b] = 0$, info$_B[4][b] = 0$, info$_B[5][b] = 0$

Check $(X, X_1, X_2)$ & Update $P$ and $W$ (if required):

If $((X_2 = \emptyset) || (X_2 = \emptyset))$ TRUE

Do nothing! //No split

If $((X_2 > \emptyset) && (X_2 > \emptyset))$

Remove $X$ from $P$.

Add $X_1$ to $P$.

Add $X_2$ to $P$.

Add $(X, X_1, X_2)$ to $W$.

After refinement with respect to $B_0$ and action $b$, $P$ and $W$ will not change.

$P = \{B_1, B_2\}$

$W = \{(B_0, B_1, B_2)\}$
Choose & remove any splitter (S) in W = \{(B_0, B_1, B_2)\} // working with Splitter (B_0, B_1, B_2)  
Case S is a compound splitter composed of (B_i, B_{i+1}) // B = B_i \cup B_{i+1}, assume 
|B_i| < |B_{i+1}|

Splitter (B_0, B_1, B_2) is a compound splitter. B = B_0, B_i = B_1, B_{i+1} = B_2

For each \( \alpha \in A \) Start with action a, \( T^{-1}_\alpha[B_0] = \{0, 1, 2\} \)

Construct \( I = \{X \mid X \in P \land X \subseteq T^{-1}_\alpha[B]\} \)
// i.e., I is a set of those blocks in P that are going to be splitted by S
// NB: P is already refined w.r.t. B.
I = {B_1}

Calculate map \( \text{info}_{B} \)
\( \text{info}_{B_1}[0][a] = 1, \text{info}_{B_1}[1][a] = 1, \text{info}_{B_1}[2][a] = 1 \)
\( \text{info}_{B_1}[3][a] = 0, \text{info}_{B_1}[4][a] = 0, \text{info}_{B_1}[5][a] = 0 \)

For each block X in I: only B_1

After refinement with respect to B_0 and action c, partition (P) will look like the following:
P = \{B_1, B_3, B_4\}; W = \{(B_0, B_1, B_2)\}
Where B_1 = \{0, 1, 2\}, B_2 = \{3, 4, 5\}, B_3 = \{3, 4\}, B_4 = \{5\}
At this point, the LTS Partition, P, should look like Fig. 1(b).
Compute \((X_1, X_2, X_3)\) as follows:
\(X_1\): Set of states in \(X\) that goes to \(B_i\) but not to \(B_{ii}\) with action \(\alpha\).
\(X_2\): Set of states in \(X\) that goes to \(B_{ii}\) but not to \(B_i\) with action \(\alpha\).
\(X_3\): Set of states in \(X\) that goes to both \(B_i\) and \(B_{ii}\).

// This 3 subblocks are calculated directly from info\(B\) and info\(B_{ii}\) as follows:
For each state \(s\) in block \(X\): // Let’s take \(s=0\) as an example
If \((\text{info}\(B_i\)[\(s\)][\(\alpha\)] == \text{info}\(B\)[\(s\)][\(\alpha\)])\) TRUE: \(\text{info}\(B_{ii}\)[\(0\)][\(\alpha\)] = 1
Add \(s\) to \(X_1\)
Add \(s=0\) to \(X_1\)
\(\text{info}\(B_{ii}\)[\(s\)][\(\alpha\)] = \text{info}\(B\)[\(s\)][\(\alpha\)]
If \((\text{info}\(B_i\)[\(s\)][\(\alpha\)] > 0) \&\&( \text{info}\(B_i\)[\(s\)][\(\alpha\)] < \text{info}\(B\)[\(s\)][\(\alpha\)])\) FALSE: ((\(\text{info}\(B_{ii}\)[\(0\)][\(\alpha\)] > 0) but \(\text{info}\(B\)[\(0\)][\(\alpha\)] == \text{info}\(B_{ii}\)[\(0\)][\(\alpha\)]\))
Add \(s\) to \(X_2\)
\(\text{info}\(B_{ii}\)[\(s\)][\(\alpha\)] = \text{info}\(B\)[\(s\)][\(\alpha\)] - \text{info}\(B_i\)[\(s\)][\(\alpha\)]
Applying the algorithm to all states in \(B_{ii}\), we get the following:
\(X_1\)={0,1,2}=\(B_{ii}\)
\(X_2\)={}
\(X_3\)={}

Check \((X_1, X_2, X_3)\) & Update \(P\) and \(W\) (if required):
If \(((X==X_1) || (X==X_2)) || (X==X_3))\) TRUE
Do nothing! //No split
Else
Replace \(X\) in \(P\) by non-null \(X_1, X_2, X_3\)
Add non-null \(X_1, X_2, X_3\) to \(W\) in the same way as in the simple splitter case, except that if all \(X_1, X_2, X_3\) are non-null add the following to \(W\):
\((X_1, X_2, X_3), (X_2, X_3, X_1)\)

After refinement with respect to \((B_0, B_1, B_2)\) and action \(\alpha\), \(P\) and \(W\) will not change.
\(P=\{B_1, B_3, B_4\}\);
\(W=\{(B_2, B_3, B_4)\}\)
Where \(B_1\)={0,1,2}, \(B_2\)={3,4,5}, \(B_3\)={3,4}, \(B_4\)={5}

Choose & remove any splitter \((S)\) in \(W\) \(W=\{(B_2, B_3, B_4)\}\) // Still working with Splitter \((B_0, B_1, B_2)\)

Case \(S\) is a compound splitter composed of \((B, B_{ii}, B_{ii})\) // \(B = B_i \cup B_{ii}\), assume \(|B_i|<|B_{ii}|\)
Still: Splitter \((B_0, B_1, B_2)\) is a compound splitter. \(B=B_0, B_i=B_1, B_{ii}=B_2\)

For each \(\alpha \in A\) Next with action \(b\), \(T_b^{-1}[B_0] = \{0,1,2\}\)
Construct \(I = \{X \mid X \in P \wedge X \subseteq T_\alpha^{-1}[B]\}\)
// i.e., I is a set of those blocks in P that are going to be splitted by S
// NB: P is already refined w.r.t. B.
l={B1}
Calculate map infoBi
infoB1[0][b]=0, infoB1[1][b]=0, infoB1[2][b]=0
infoB1[3][b]=0, infoB1[4][b]=0, infoB1[5][b]=0
For each block X in I: only B1
Compute (X1,X2,X3) as follows:
  X1: Set of states in X that goes to Bi but not to Bii with action a.
  X2: Set of states in X that goes to Bii but not to Bi with action a.
  X3: Set of states in X that goes to both Bi and Bii.
// This 3 subblocks are calculated directly from infoB and infoB1 as follows:
For each state s in block X:
  If (infoBi[s][a]== infoB[s][a])
    Add s to X1
    infoB1[s][a]=0
  If (infoBi[s][a]== 0)
    Add s to X2
    infoB1[s][a]=infoB[s][a]
  If ((infoBi[s][a]> 0) &&( infoB1[s][a]< infoB[s][a] ))
    Add s to X3
    infoB1[s][a]=infoB[s][a]- infoB1[s][a]
Applying the algorithm to all states in B1, we get the following:
X1={}
X2={0,1,2}=B1
X3={}
Check (X1,X2,X3) & Update P and W (if required):
  If ((X==X1) || (X==X2) || (X==X3)) TRUE
    Do nothing! //No split
  Else
    Replace X in P by non-null X1,X2,X3
    Add non-null X1,X2,X3 to W in the same way as in the simple splitter case, except that if all X1,X2,X3 are non-null add the following to W:
    (X1,X2,X3),(X23,X23,X3)
After refinement with respect to (B0,B1,B2) and action a, P and W will not change.
P={B1,B3, B4};
W={(B2,B3, B4)}
Where B1={0,1,2}, B2={3,4,5}, B3={3,4}, B4={5}
Choose & remove any splitter \((S)\) in \(W = \{(B_2, B_3, B_4)\} \) // Still working with Splitter \((B_0, B_1, B_2)\)

Case \(S\) is a compound splitter composed of \((B, B_i, B_{ii})\) // \(B = B_i \cup B_{ii}\), assume

\[|B_i| < |B_{ii}|\]

Still: Splitter \((B_0, B_1, B_2)\) is a compound splitter. \(B = B_0, B_i = B_1, B_{ii} = B_2\)

For each \(\alpha \in A\) Next with action \(c\), \(T_c^{-1}[B_0] = \{3, 4\}\)

Construct \(I = \{X \mid X \in P \wedge X \subseteq T_c^{-1}[B]\}\)

// i.e., \(I\) is a set of those blocks in \(P\) that are going to be splitted by \(S\)
// NB: \(P\) is already refined w.r.t. \(B\).

\(I = \{B_3\}\)

Calculate map \(\text{info}_{B_i}\)

\(\text{info}_{B_1}[0][c] = 0, \text{info}_{B_1}[1][c] = 0, \text{info}_{B_1}[2][c] = 0\)

\(\text{info}_{B_1}[3][c] = 0, \text{info}_{B_1}[4][c] = 0, \text{info}_{B_1}[5][c] = 0\)

For each block \(X\) in \(I\): only \(B_3\)

Compute \((X_1, X_2, X_3)\) as follows:

\(X_1\): Set of states in \(X\) that goes to \(B_i\) but not to \(B_{ii}\) with action \(\alpha\).

\(X_2\): Set of states in \(X\) that goes to \(B_{ii}\) but not to \(B_i\) with action \(\alpha\).

\(X_3\): Set of states in \(X\) that goes to both \(B_i\) and \(B_{ii}\).

// This 3 subblocks are calculated directly from \(\text{info}_B\) and \(\text{info}_{B_i}\) as follows:

For each state \(s\) in block \(X\):

- If \((\text{info}_{B_i}[s][\alpha] == \text{info}_B[s][\alpha])\)
  - Add \(s\) to \(X_1\)

- If \((\text{info}_{B_i}[s][\alpha] == 0)\)
  - Add \(s\) to \(X_2\)

- If \((\text{info}_{B}[s][\alpha] > 0) \&\& (\text{info}_{B_i}[s][\alpha] < \text{info}_B[s][\alpha])\)
  - Add \(s\) to \(X_3\)

\(\text{info}_{B_1}[s][\alpha] = \text{info}_B[s][\alpha] - \text{info}_{B_i}[s][\alpha]\)

Applying the algorithm to all states in \(B_3\), we get the following:

\(X_1 = \{\}\\)

\(X_2 = \{3, 4\} = B_3\)

\(X_3 = \{\}\)

Check \((X_1, X_2, X_3)\) & Update \(P\) and \(W\) (if required):

- If \((\text{size}X_1 || \text{size}X_2 || \text{size}X_3)\) \(\text{TRUE}\)
  - Do nothing! // No split

- Else
  - Replace \(X\) in \(P\) by non-null \(X_1, X_2, X_3\)
  - Add non-null \(X_1, X_2, X_3\) to \(W\) in the same way as in the simple splitter case, except that if all \(X_1, X_2, X_3\) are non-null add the following to \(W\):
    \((X_1, X_2, X_3), (X_2, X_3, X_1)\)

After refinement with respect to \((B_0, B_1, B_2)\) and action \(a\), \(P\) and \(W\) will not change.

\(P = \{B_1, B_3, B_4\}\); \(W = \{(B_2, B_3, B_4)\}\)

Where \(B_1 = \{0, 1, 2\}, B_2 = \{3, 4, 5\}, B_3 = \{3, 4\}, B_4 = \{5\}\)
Finally:

Choose & remove any splitter (S) in W

We can easily see that, after refinement with respect to (B_2,B_3,B_4) and all actions a, b, and c, P and W will not change.

P=\{B_1,B_3,B_4\};
W=\{(B_2,B_3,B_4)\}

Where B_1=\{0,1,2\}, B_2=\{3,4,5\}, B_3=\{3,4\}, B_4=\{5\}

At this point, the LTS Partition, P, should look like Fig. 1(b). Proceeding from Fig. 1(b), the algorithm will generate a minimized LTS as in Fig. 1(c). This is done by replacing each block by a new state in the minimized graph.

References:


