Improving an Interactive Visualization of Transition Systems

B. Ploeger    C. Tankink
Eindhoven University of Technology

November 13, 2008
ProSe
Eindhoven University of Technology
Introduction

Transition system

- Consists of states and transitions
- Describes behaviour of a software system or model
- Underlies state charts, MSCs, process algebras, etc.

System analysis

Does the system meet its specification / requirements?

Use formal verification methods to check:
- Presence of desirable behaviour (e.g. liveness)
- Absence of undesirable behaviour (e.g. deadlocks)
Introduction

Transition system
- Consists of states and transitions
- Describes behaviour of a software system or model
- Underlies state charts, MSCs, process algebras, etc.

System analysis
- Does the system meet its specification / requirements?
- Use formal verification methods to check:
  - Presence of desirable behaviour (e.g. liveness)
  - Absence of undesirable behaviour (e.g. deadlocks)
Introduction

Visualization of Transition systems

- Formal methods are powerful, but:
  - Require domain-specific knowledge and experience
  - Do not allow to “explore” the behaviour
Introduction

Visualization of Transition systems

- Formal methods are powerful, but:
  - Require domain-specific knowledge and experience
  - Do not allow to “explore” the behaviour
- Use visualization techniques to:
  - Provide an **overview** of the transition system
  - Show **symmetry** in the transition system
  - Provide **interaction** with the transition system
Introduction

Visualization technique [Van Ham et al., 2000 & 2002]

- Cluster states based on structural properties
- Interrelate clusters: ancestor – descendants
- Visualize tree in 3D, similar to cone trees
- Add interactive features
# Introduction

**Visualization technique [Van Ham et al., 2000 & 2002]**

- Cluster states based on structural properties
- Interrelate clusters: ancestor – descendants
- Visualize tree in 3D, similar to cone trees
- Add interactive features

## Our contribution

- Solve 6 problems identified by users
- Add 2 interactive features:
  - Simulation
  - Visualization of state attribute values
Original approach
Step 1: rank states

Minimal distance from initial state
Original approach

Step 2: cluster states

Put two states in same cluster if they:

- Have equal ranks, and
- Can both reach some state at equal or higher rank
Original approach
Step 3: position clusters

- Assign clusters to planes
Original approach
Step 3: position clusters

Descendants

$D$ is descendant of $C$ if:

- $D$ is at one rank higher than $C$
- A state in $C$ has a transition to a state in $D$
Original approach
Step 3: position clusters

Descendants

$D$ is descendant of $C$ if:
- $D$ is at one rank higher than $C$
- A state in $C$ has a transition to a state in $D$

Positioning descendants

- Some descendants are centered below $C$ (heuristic)
- All others are placed on a circle below $C$
Original approach

Step 3: position clusters

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
Original approach
Step 3: position clusters

Heuristic for centering descendants below cluster $C$

1. If a unique \textit{largest} descendant exists, center it below $C$
2. If a unique \textit{smallest} descendant exists, center it below $C$, unless it has descendants itself and step 1 centered a descendant already
Original approach
Step 3: position clusters

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless it has descendants itself and step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.
Original approach
Step 3: position clusters
Original approach
Step 3: position clusters

D1  D2  D3  D4  D5

C

D5
Original approach

Step 3: position clusters

D1  D2  D3  D4  D5

C

D5
Original approach

Step 3: position clusters
Original approach

Step 3: position clusters

Cluster radius and subtree size

- Radius of cluster: depends linearly on number of states
- Size of cluster (subtree): radius of bounding cylinder containing entire subtree.
Original approach
Step 3: position clusters

Cluster radius and subtree size
- Radius of cluster: depends \textit{linearly} on number of states
Original approach
Step 3: position clusters

Cluster radius and subtree size

- Radius of cluster: depends \textit{linearly} on number of states
- Size of cluster (subtree): radius of \textit{bounding cylinder} containing entire subtree.
Original approach

Step 4: connect clusters

Connect cluster with its descendants using a cone
Original approach
Step 4: connect clusters

Connect cluster with its descendants using a cone
Problem 1
Descendants may overlap

<table>
<thead>
<tr>
<th>Heuristic for centering descendants below cluster $C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. If a unique largest descendant exists, center it below $C$</td>
</tr>
<tr>
<td>2. If a unique smallest descendant exists, center it below $C$, unless it has descendants itself and step 1 centered a descendant already</td>
</tr>
<tr>
<td>3. If there is only 1 descendant left, undo step 1.</td>
</tr>
</tbody>
</table>
Problem 1
Descendants may overlap

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless it has descendants itself and step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.

- Two clusters may be centered
- These clusters then overlap, which impairs the overview
Problem 1
Descendants may overlap

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless it has descendants itself and step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.

- Two clusters may be centered
- These clusters then overlap, which impairs the overview
- Solution: weaken the condition in step 2
Problem 1
Descendants may overlap

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.

- Two clusters may be centered
- These clusters then overlap, which impairs the overview
- Solution: weaken the condition in step 2
Improvement 1
Prevent overlapping descendants

Original approach
Improvement 1
Prevent overlapping descendants

Original approach

Improvement
Problem 2
Main behaviour is not always centered

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$.
2. If a unique smallest descendant exists, center it below $C$, unless step 1 centered a descendant already.
3. If there is only 1 descendant left, undo step 1.
Problem 2
Main behaviour is not always centered

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.

- Largest descendant usually contains “normal” behaviour
- More intuitive if this is always kept centered
Problem 2
Main behaviour is not always centered

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless step 1 centered a descendant already
3. If there is only 1 descendant left, undo step 1.

- Largest descendant usually contains “normal” behaviour
- More intuitive if this is always kept centered
- Solution: remove step 3
Problem 2
Main behaviour is not always centered

Heuristic for centering descendants below cluster $C$

1. If a unique largest descendant exists, center it below $C$
2. If a unique smallest descendant exists, center it below $C$, unless step 1 centered a descendant already

- Largest descendant usually contains “normal” behaviour
- More intuitive if this is always kept centered
- Solution: remove step 3
Improvement 2
Keep largest descendant centered

Original approach
Improvement 2
Keep largest descendant centered

Original approach

Improvement
Problem 3
No method for positioning non-centered descendants

- Non-centered descendants are not positioned “smartly”
- Doing so can improve symmetry
Problem 3
No method for positioning non-centered descendants

- Non-centered descendants are not positioned “smartly”
- Doing so can improve symmetry
- Solution: assign positions based on descendants’ sizes (heuristic)
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3

Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3

Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants
Improvement 3
Heuristic for positioning non-centered descendants

Original approach

Improvement
Problem 4

Relative sizes of clusters are misleading

- Cluster radius depends linearly on number of states
- All states fit on cluster’s boundary
Problem 4
Relative sizes of clusters are misleading

- Cluster radius depends linearly on number of states
- All states fit on cluster’s boundary

Problem
- Clusters seem much larger than they are
- Misinterpretation of relative numbers of states
Problem 4
Relative sizes of clusters are misleading

- Cluster radius depends linearly on number of states
- All states fit on cluster’s boundary

Problem
- Clusters seem much larger than they are
- Misinterpretation of relative numbers of states

Solution
- Keep cluster’s area proportional to number of states
- Cluster radius depends on square root of this number
Improvement 4

Show number of states by cluster area

Original approach
Improvement 4
Show number of states by cluster area

Original approach

Top cluster: 1 state
Bottom cluster: 26 states
Improvement 4
Show number of states by cluster area

- **Original approach**
  - Top cluster: 1 state
  - Bottom cluster: 26 states

- **Improvement**
Problem 5
Cluster size disregards depth of subtree

- Cluster size is used in positioning heuristics
- Radius of bounding cylinder containing entire subtree
Problem 5
Cluster size disregards depth of subtree

- Cluster size is used in positioning heuristics
- **Radius** of bounding cylinder containing entire subtree
- Problem: these clusters have the same size:

![Diagram showing two clusters of different shapes with the same size indicated by the radius of the bounding cylinder.](image)
Problem 5
Cluster size disregards depth of subtree

- Cluster size is used in positioning heuristics
- **Radius** of bounding cylinder containing entire subtree
- Problem: these clusters have the same size:

![Cluster diagrams]

- Solution: define size as **volume** of bounding cylinder
Improvement 5
Use subtree volume for cluster size

Original approach
Improvement 5
Use subtree volume for cluster size

Original approach

Improvement
Problem 6
Large connecting cones are misleading

- Connect cluster with descendants using a single cone
- Cone can become very large
- Misleading: large objects seem to contain lots of states
Problem 6
Large connecting cones are misleading

- Connect cluster with descendants using a single cone
- Cone can become very large
- Misleading: large objects seem to contain lots of states
- Solution: connect using several small cones
Problem 6
Large connecting cones are misleading

- Connect cluster with descendants using a single cone
- Cone can become very large
- Misleading: large objects seem to contain lots of states
- Solution: connect using several small cones
Improvement 6
Connect descendants using small cones

Original approach
Improvement 6
Connect descendants using small cones

Original approach

Every cluster: 1 state
Improvement 6
Connect descendants using small cones

Original approach
Improvement

Every cluster: 1 state
Interactive simulation

- Step through state space
- Most simulation tools: text-based
- Hard to maintain mental map of visited states
- Easier when visualized
Interactive simulation
Interactive simulation
Visualizing state attribute values

- Every state is a combination of \textit{values}
- One for every variable/parameter in the system
Visualizing state attribute values

- Every state is a combination of values.
- One for every variable/parameter in the system.

Marking states and clusters

- Specify **mark rules**: certain parameter must (not) have certain value(s).

- Specify how to combine mark rules:
  - match any: mark state if any mark rule is matched
  - match all: mark state if all mark rules are matched
  - match separately: mark state separately for every rule

A cluster is marked if any of its states is marked.
Visualizing state attribute values

- Every state is a combination of values
- One for every variable/parameter in the system

Marking states and clusters

- Specify mark rules: certain parameter must (not) have certain value(s)
- Specify how to combine mark rules:
  - *match any*: mark state if any mark rule is matched
  - *match all*: mark state if all mark rules are matched
  - *match separately*: mark state separately for every rule
Visualizing state attribute values

- Every state is a combination of values
- One for every variable/parameter in the system

Marking states and clusters

- Specify **mark rules**: certain parameter must (not) have certain value(s)
- Specify how to combine mark rules:
  - *match any*: mark state if any mark rule is matched
  - *match all*: mark state if all mark rules are matched
  - *match separately*: mark state separately for every rule
- A cluster is marked if **any** of its states is marked
Example: 5 dining philosophers
Mark rules:

- Fork 1 is picked up by philosopher 2
- Fork 2 is picked up by philosopher 2 or 3
- Fork 3 is picked up by philosopher 3
Mark rules:

- Fork 1 is picked up by philosopher 2
- Fork 2 is picked up by philosopher 2 or 3
- Fork 3 is picked up by philosopher 3

Match any
Mark rules:

- Fork 1 is picked up by philosopher 2
- Fork 2 is picked up by philosopher 2 or 3
- Fork 3 is picked up by philosopher 3

Match all
Mark rules:

- **Fork 1** is picked up by philosopher 2
- **Fork 2** is picked up by philosopher 2 or 3
- **Fork 3** is picked up by philosopher 3

Match separately
Mark rules:

- Fork 1 is picked up by philosopher 2
- Fork 2 is picked up by philosopher 2 or 3
- Fork 3 is picked up by philosopher 3

Match separately
About LTSView

- Distributed as part of the mCRL2 toolset
- Analysis and verification of distributed systems and protocols
- Modelling language: process algebra
- Linux, Unix, Mac OS X, Windows
- Boost software license
- http://www.mcrl2.org
Concluding remarks

Visualization of transition systems

- **Aim:** to provide an overview of system behaviour
- **Six improvements to existing technique:** improve overview and symmetry
- **Illustrated by real-life examples**
- **Added two interactive features:** simulation and marking

Possible future work
- Improve visualization for larger systems (scalability)
- Investigate other clustering techniques (on-the-fly)
- Positioning of states within cluster
Concluding remarks

Visualization of transition systems

- Aim: to provide an overview of system behaviour
- Six improvements to existing technique: improve overview and symmetry
- Illustrated by real-life examples
- Added two interactive features: simulation and marking

Possible future work

- Improve visualization for larger systems (scalability)
- Investigate other clustering techniques (on-the-fly)
- Positioning of states within cluster
Thank you!

(Hint: read today’s Cursor!)