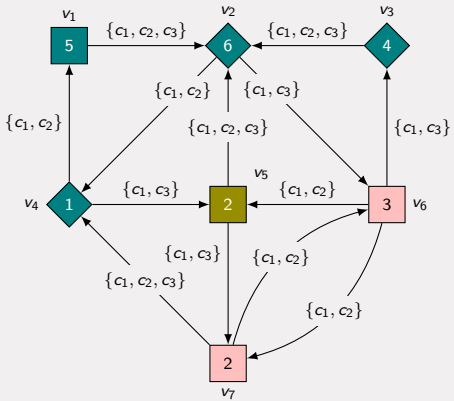
Verifying SPLs using parity games expressing variability



Sjef van Loo

6 November, 2019

Msc Thesis
Computer Science and Engineering

Supervised by T.A.C. Willemse

TU/e

Outline

- Verification & SPLs
- Problem statement
- Variability Parity Games
- VPG algorithms
- Experimental results

Verification & SPLs

- Formally verify software by modelling its behaviour and expressing a requirement
- SPLs describe multiple software products, it does so by using features
- TODO: LTS en FTS uitleggen? Wellicht noemen maar niet uitleggen...

Problem statement

- Verify all the products in an SPL
- Do so more efficiently than verifying them independently

Variability parity game

- Explain Parity Games using an example
- What does solving a PG mean
- PGs can be used to check software products

- Explain VPGs using an example
- What does solving a VPG mean
- VPGs can be used to check SPLs

VPG algorithms - Recursive algorithm

- The recursive algorithm reasons about sets of vertices
- Attractor calculation example

- the recursive algorithm for VPGs reasons about sets of vertex configuration pairs
- Attractor calculation example on VPG
- Function-wise representation to efficiently perform attractor calcs
- Short explanation of symbolic representation
- Time complexities

VPG algorithms - Incremental pre-solve algorithm

- Introduce algorithm
- Introduce pessimistic PGs
- We need an alg to solve PGs using pre-solved vertices for efficiency

VPG algorithms - Incremental pre-solve algorithm

- FPIte, show FP formula
- Show modified FP formula
- Explain the efficiency gained
- Very short explanation of a fixed-point

VPG algorithms - Local solving

- explain local solving
- introduced local algs for the novel VPG algs and existing PG algs.

Experimental results - SPL games

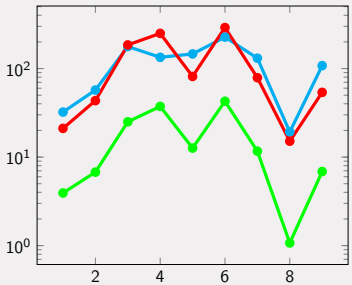


Figure: Running times, in ms, on the minepump games.

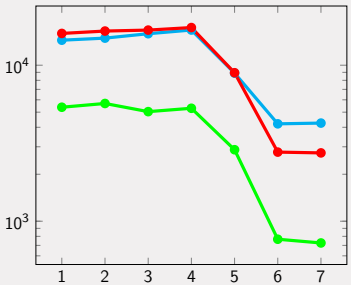


Figure: Running times, in ms, on the elevator games.

- Recursive algorithm for parity games
- Recursive algorithm for VPGs with a symbolic representation of configurations
- Recursive algorithm for VPGs with an explicit representation of configurations

Experimental results - SPL games

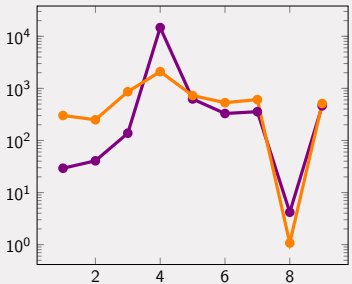


Figure: Running times, in ms, on the minepump games.

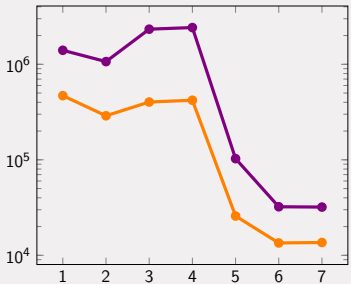


Figure: Running times, in ms, on the elevator games.

Fixed-point iteration algorithm for parity games
Incremental pre-solve algorithm

Experimental results - Random games

- Show the type of games where recursive symbolic fails and the explicit does not.

Experimental results - Local solving

- Show the same graphs but with local solving as well

Conclusions

- Collective approach can improve SPLs verifying performance
- The symbolic recursive can do this well
- The explicit recursive is "robust"
- Local solving can increase performance, however very dependent on alg & type of VPG