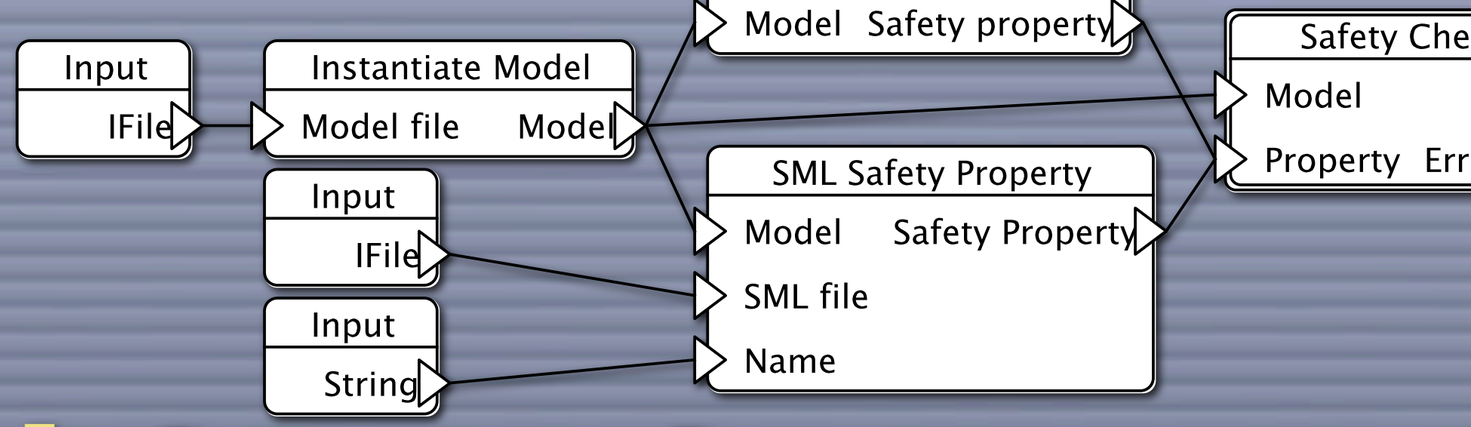
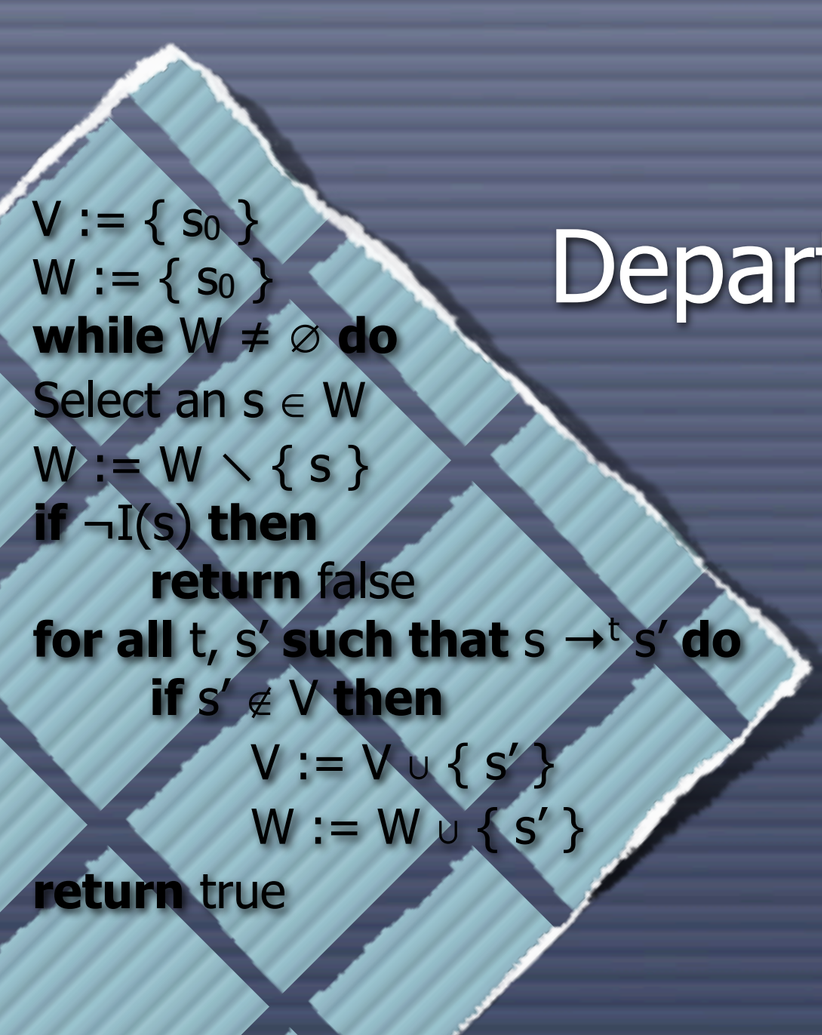
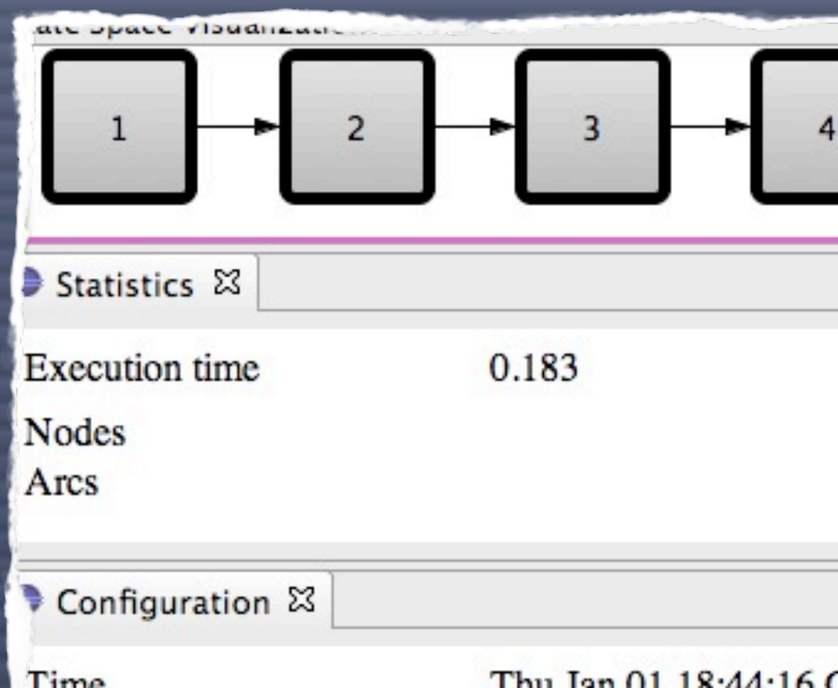


Advanced State Space Methods and ASAP: Status and Outlook

Michael Westergaard
Department of Computer Science
Aarhus University
mw@cs.au.dk





Current Status

- ASAP supports an extensive list of methods in a coherent way
- ASAP uses the language JoSEL to compose verification jobs
- New methods can be added via plug-ins

Current New Features

- Rewritten JoSEL editor with full support for hierarchical JoSEL specifications
- Support for off-line (in addition to on-the-fly) analysis of safety properties
- Off-line drawing of graphs and error traces
- Support for calculation of SCC graphs
- (32 bit) simulator that runs on 64 bit Windows
  - This is being incorporated into CPN Tools

Short-term New Features

- Even faster analysis
- Support for timed models
- Standard report
- On-line drawing of graph
- Advanced off-line visual graph reduction based on image resizing
- (On-the-fly) LTL model checking

Even Faster Analysis

- CPN Tools and ASAP currently uses SML/NJ
- In the labs we have ported parts of the simulator to SML compiler, MLton
- MLton can cut runtime down to 50% - 70%
- MLton does **not** allow dynamic code generation and you must ask all questions in advance
  - We seek a reasonable way to use SML/NJ in the initial phases (interactive investigation) and MLton for hardcore number crunching in later phases

Even Faster Analysis

- MLton provides a much better profiler than SML/NJ, leading to new insights
  - We use 30% - 50% of the time generating random numbers to choose bindings fairly
  - After eliminating the above, we use 20% - 30% of the time converting between multi-set representations

Ev…s

When doing state space analysis, we need to investigate **all** bindings, so we do not care about a fair order

- I… filer than SML/NJ, leading to new insights

We use 30% - 50% of the time generating random numbers to choose bindings fairly

After eliminating the above, we use 20% - 30% of the time converting between multi-set representations

Even Faster Analysis

- MLton provides a much better profiler than SML/NJ, leading to new insights
  - We use 30% - 50% of the time generating random numbers to choose bindings fairly
  - After eliminating the above, we use 20% - 30% of the time converting between multi-set representations

Even Faster Analysis

- MLton provides a much better profiler than SML
- W ge bi
- After eliminating the above, we use 20% - 30% of the time converting between multi-set representations

Most likely it is possible to just use the one of the state space tool when not doing simulation

Even Faster Analysis

- MLton provides a much better profiler than SML/NJ, leading to new insights
  - We use 30% - 50% of the time generating random numbers to choose bindings fairly
  - After eliminating the above, we use 20% - 30% of the time converting between multi-set representations

Even Faster Analysis

- MLton provides a much better profiler than SML/NJ, leading to new insights
- We use 30% - 50% of the time generating random numbers to choose bindings fairly
- After eliminati
- 30% of the t
multi-set repre

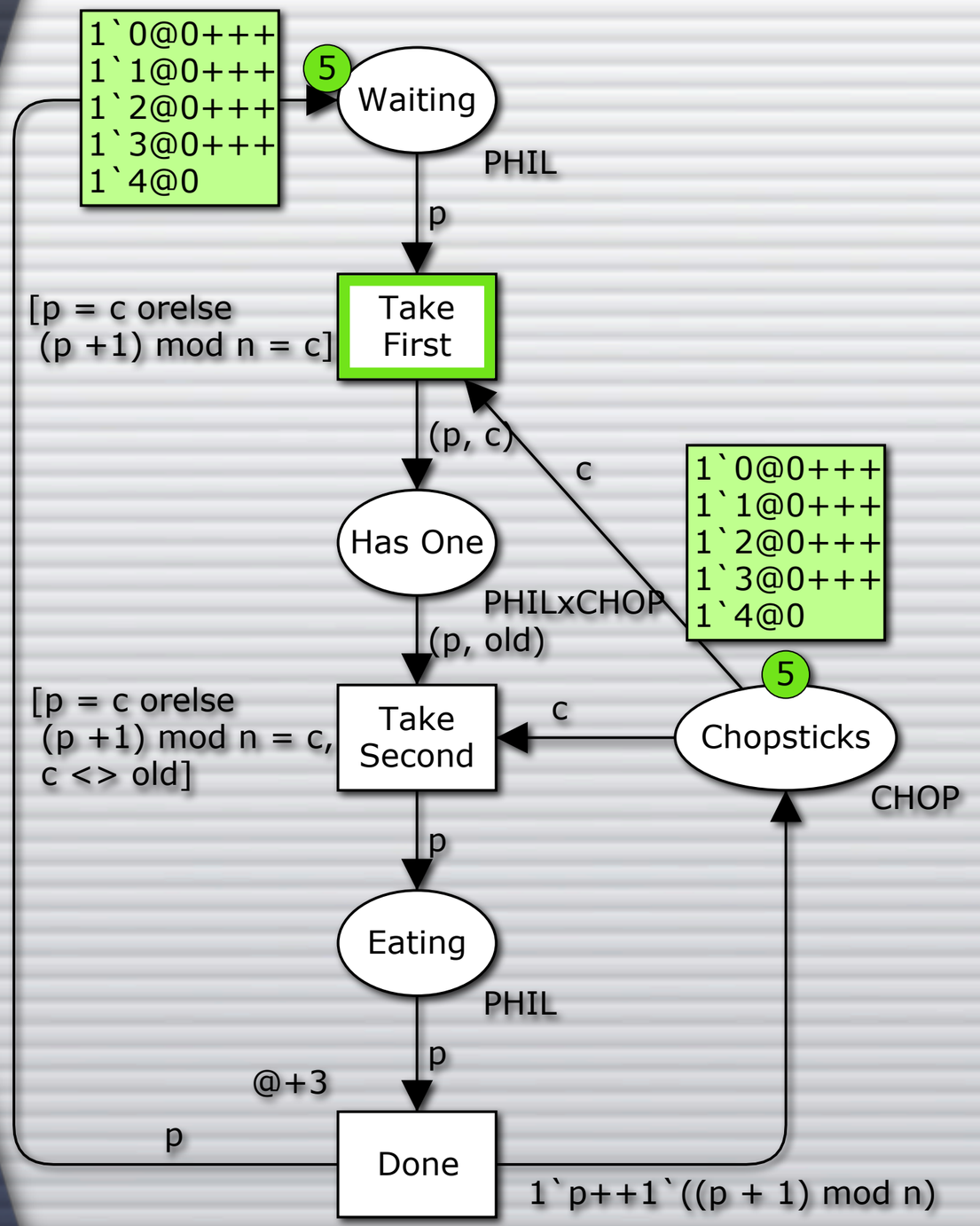
Using all of this information has allowed us to speed the tool up by a factor 6-8 (compared to the speed we have seen today)

Support for Timed Models

- We need to add time information to the state descriptor (easy)
- We need to make all the utility functions understand the time value (hash-functions, serializations, etc.)
  (easy but mind numbingly dull)
- We need to implement time equivalence (easy)

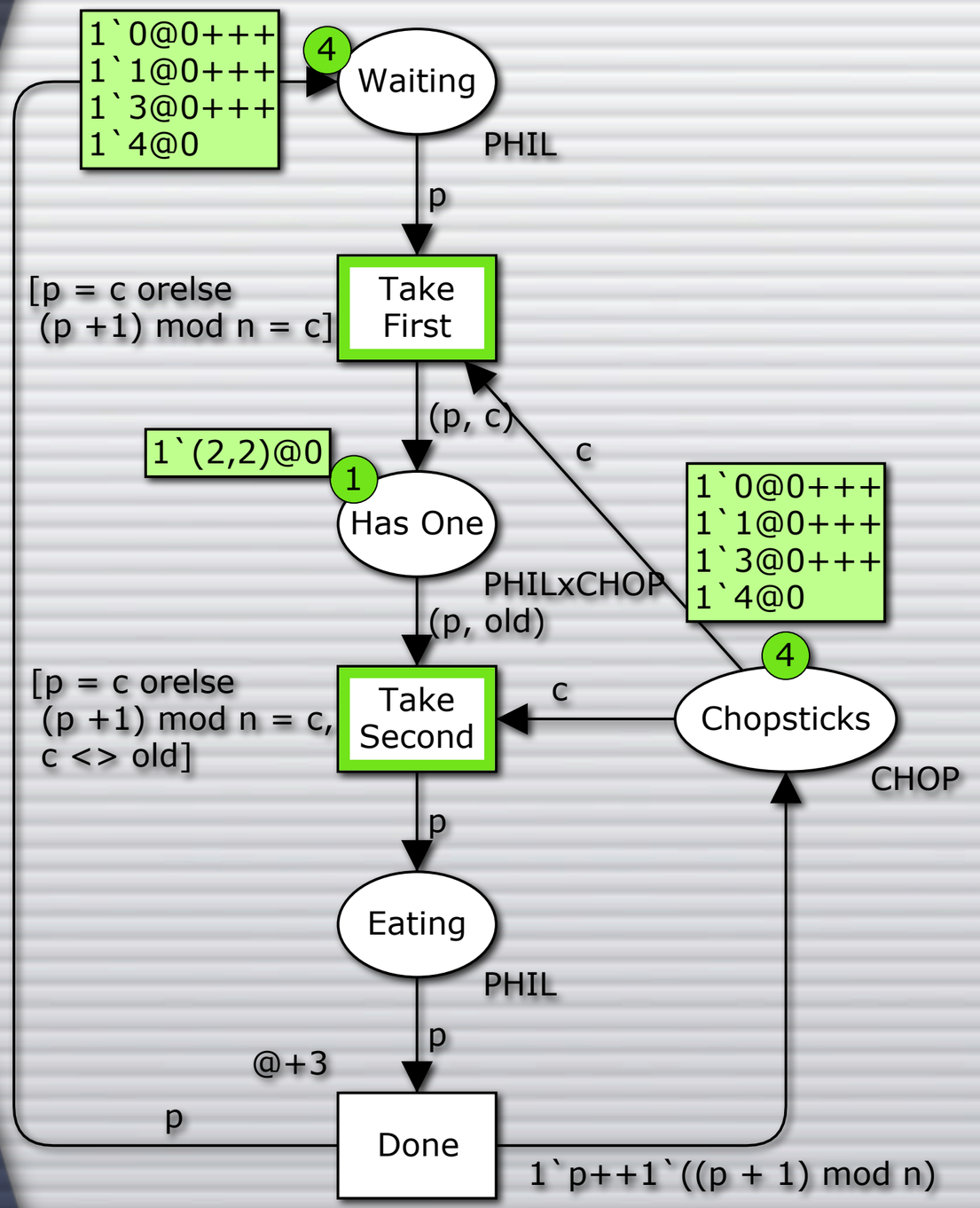
Support for Timed Models

- Models with finite untimed behavior can have infinite timed state spaces
- Time equivalence does not record absolute time stamps, only the difference between the current time and the time stamp



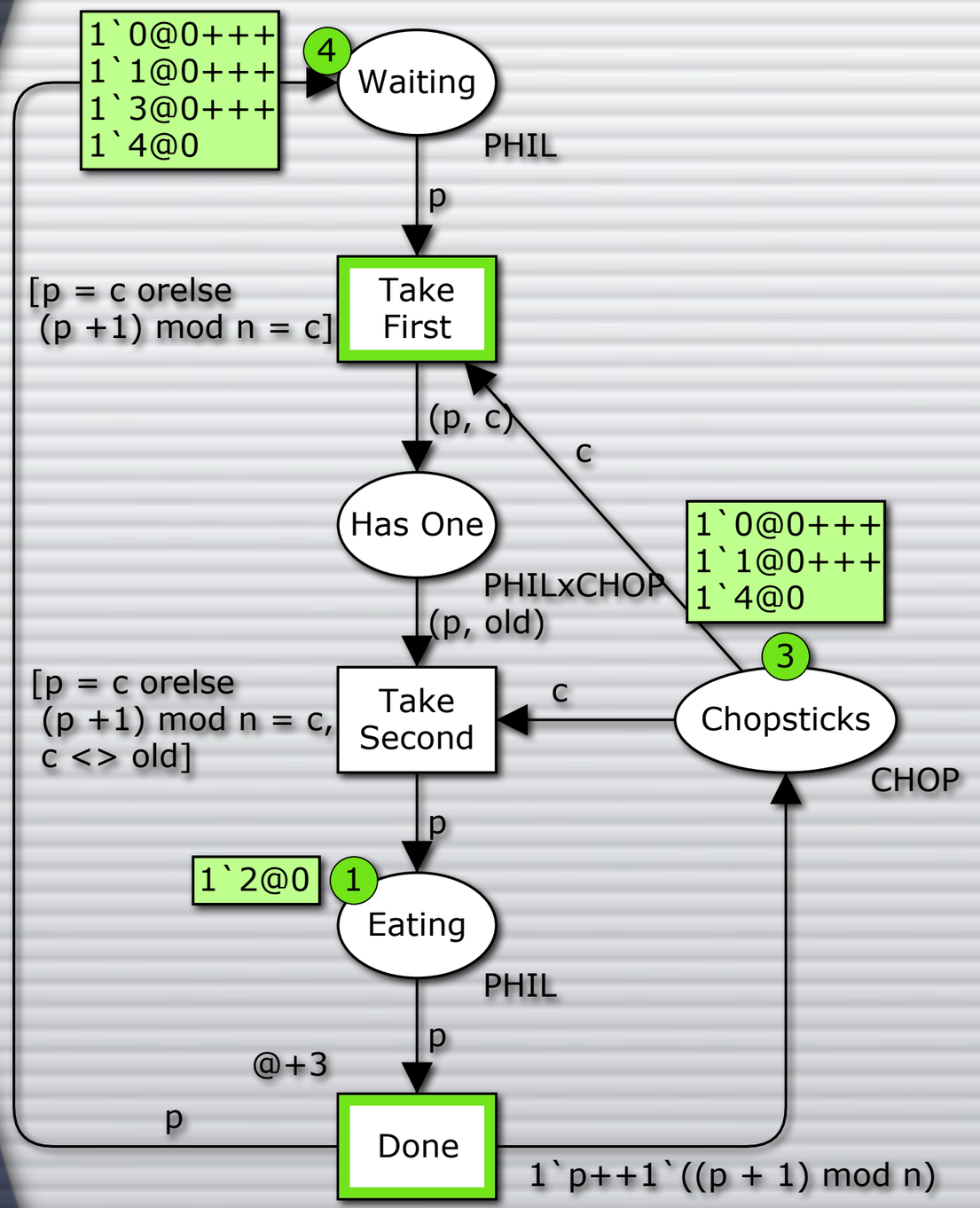
Support for Timed Models

- Models with finite untimed behavior can have infinite timed state spaces
- Time equivalence does not record absolute time stamps, only the difference between the current time and the time stamp



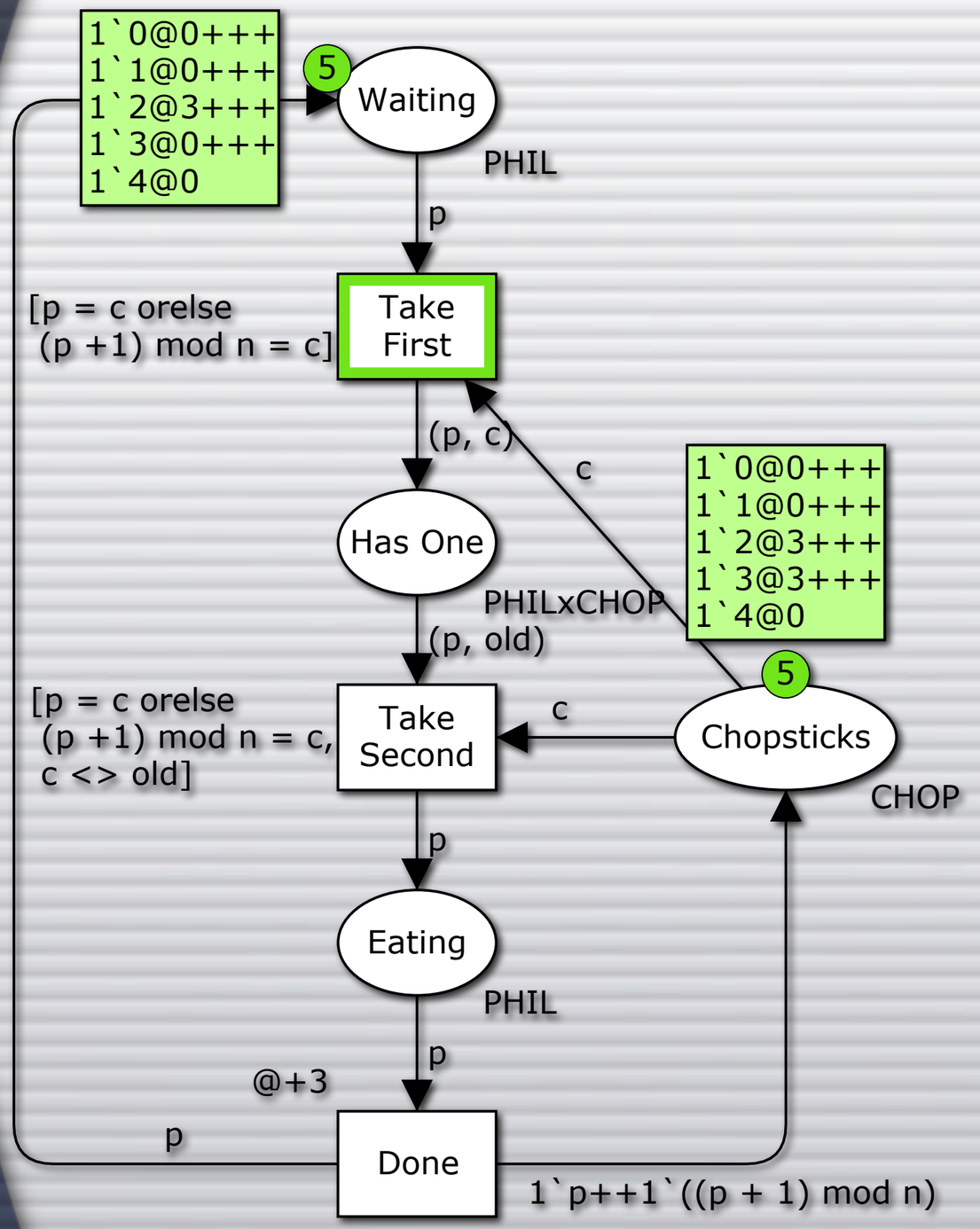
Support for Timed Models

- Models with finite untimed behavior can have infinite timed state spaces
- Time equivalence does not record absolute time stamps, only the difference between the current time and the time stamp



Support for Timed Models

- Models with finite untimed behavior can have infinite timed state spaces
- Time equivalence does not record absolute time stamps, only the difference between the current time and the time stamp



Standard Report

- Most of the engine is there (exploration, SCC graphs, extensible reporting engine)
- The properties need to be generated from the model, and code needs to be written for this

On-line Drawing of Graphs

- Now, graphs are drawn all at once
- Normally, we'll just want to explore parts of the graph interactively (like in CPN Tools)

On-line Drawing of Graphs

- ASAP keeps a real representation of the graph fragments it draws in memory (instead of just state numbers)
- If we only require to be able to draw out-going nodes, we do not even have to precompute the entire graph

On-line Drawing of Graphs

On-line Drawing of Graphs

On-line Drawing of Graphs

Display successors

Display predecessors

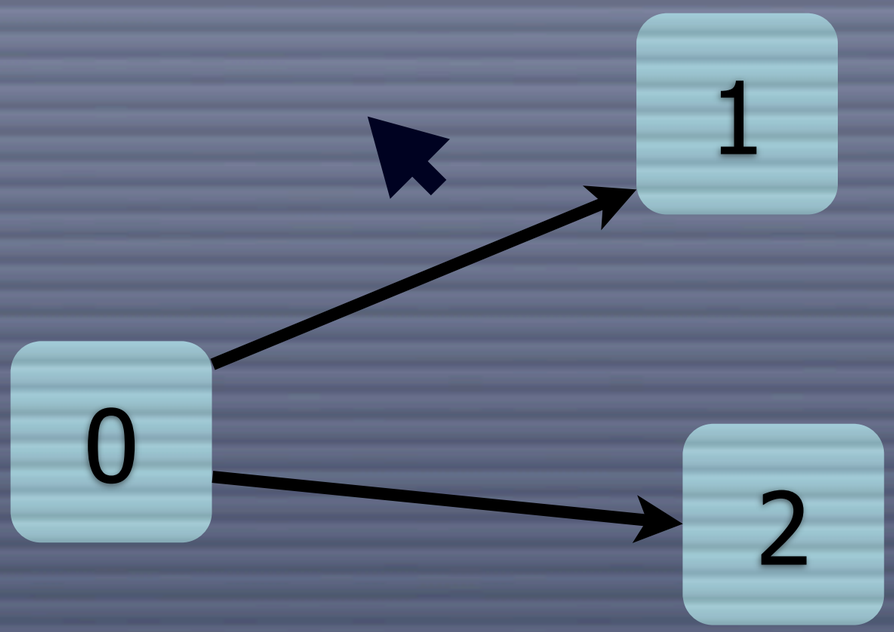
On-line Drawing of Graphs

On-line Drawing of Graphs

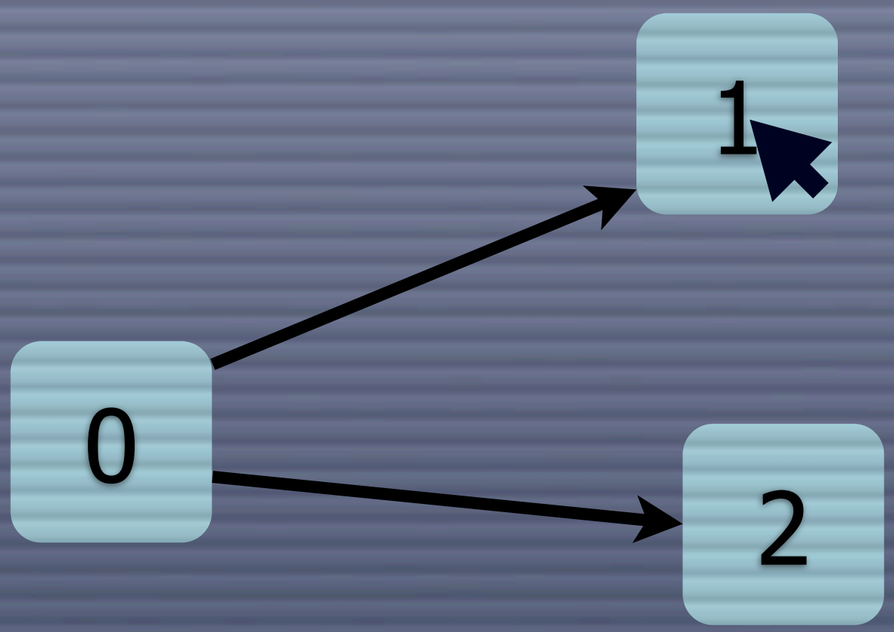
0

A representation of state 0 is sent to state space engine, which calculates and returns the successors

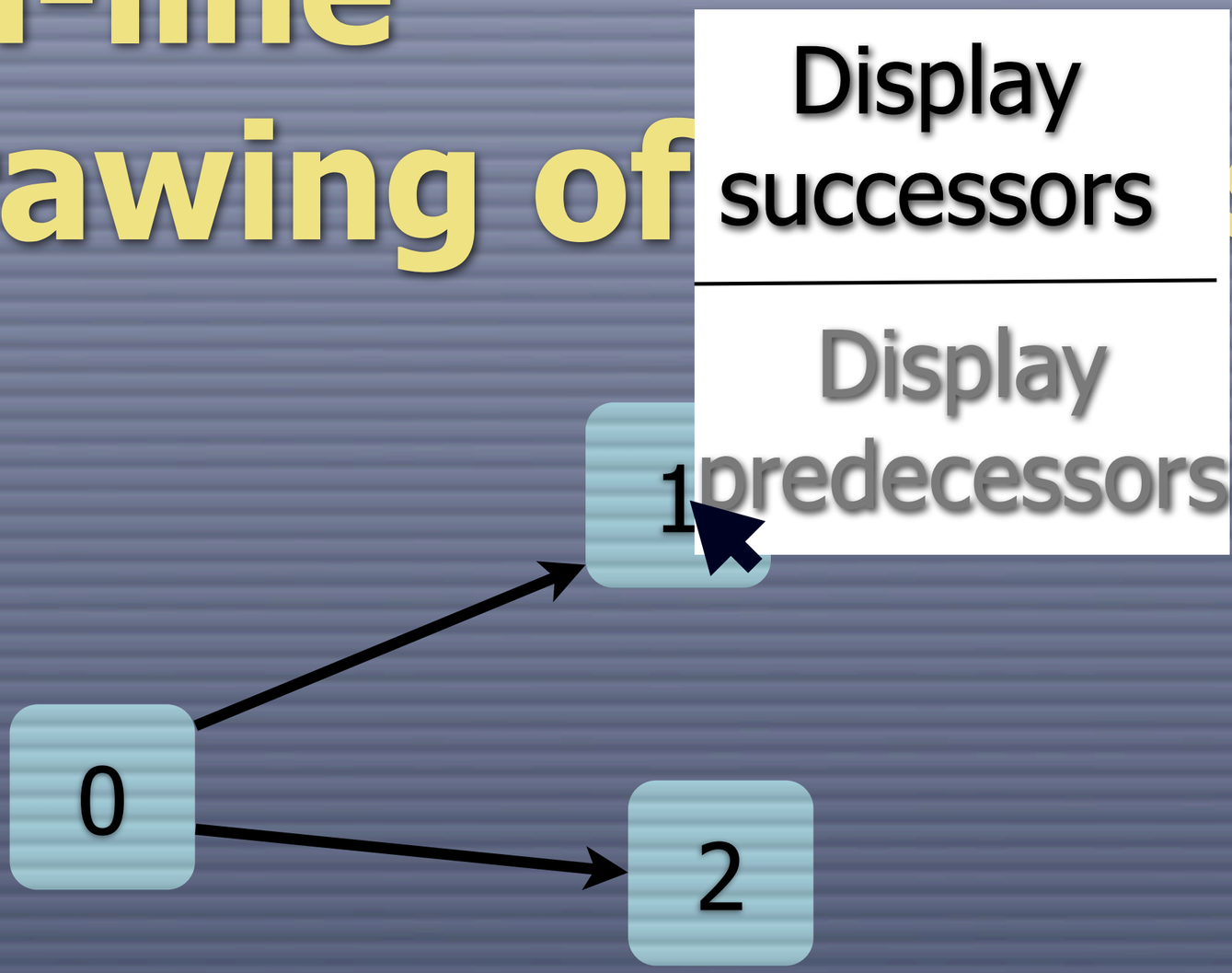
On-line Drawing of Graphs



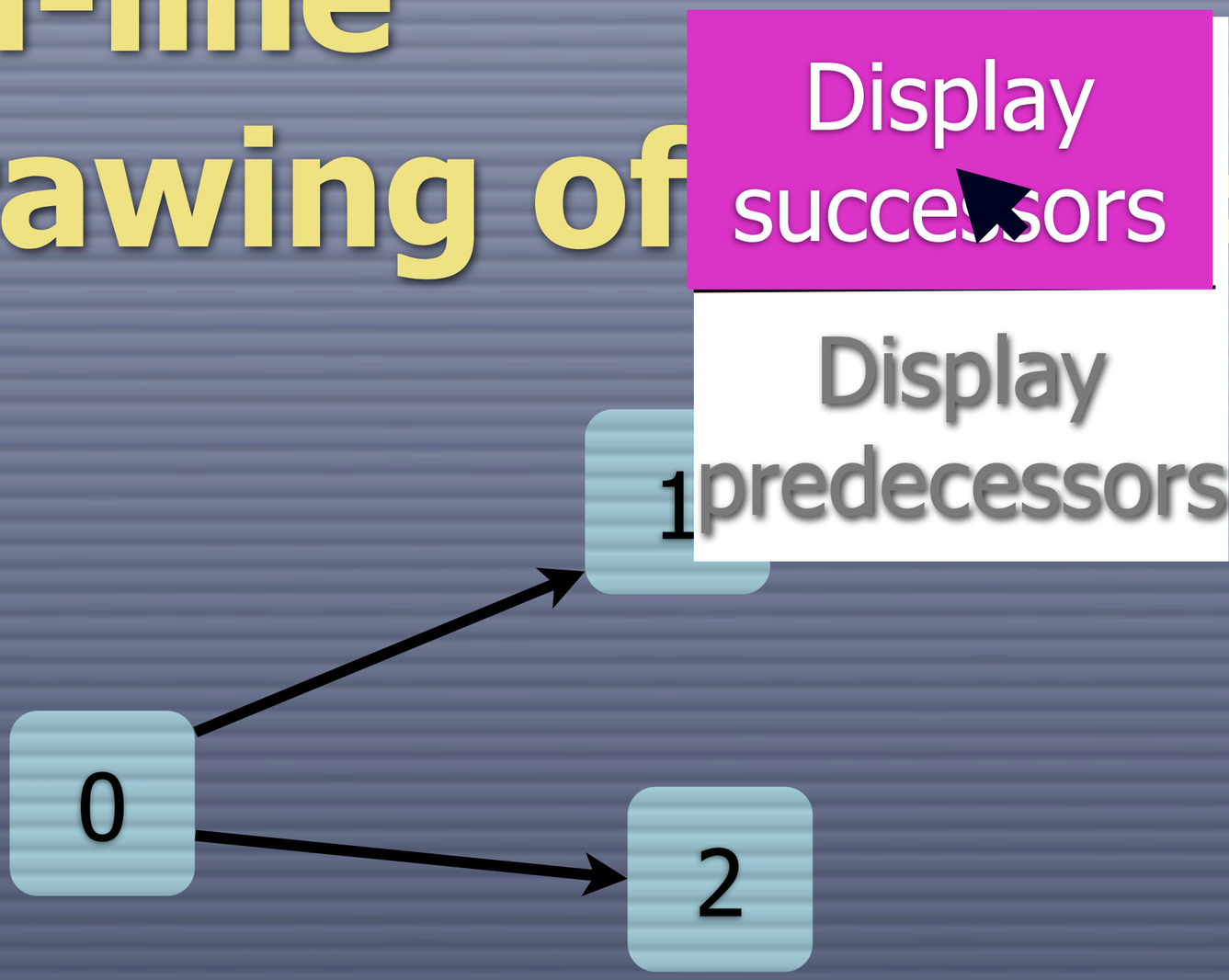
On-line Drawing of Graphs



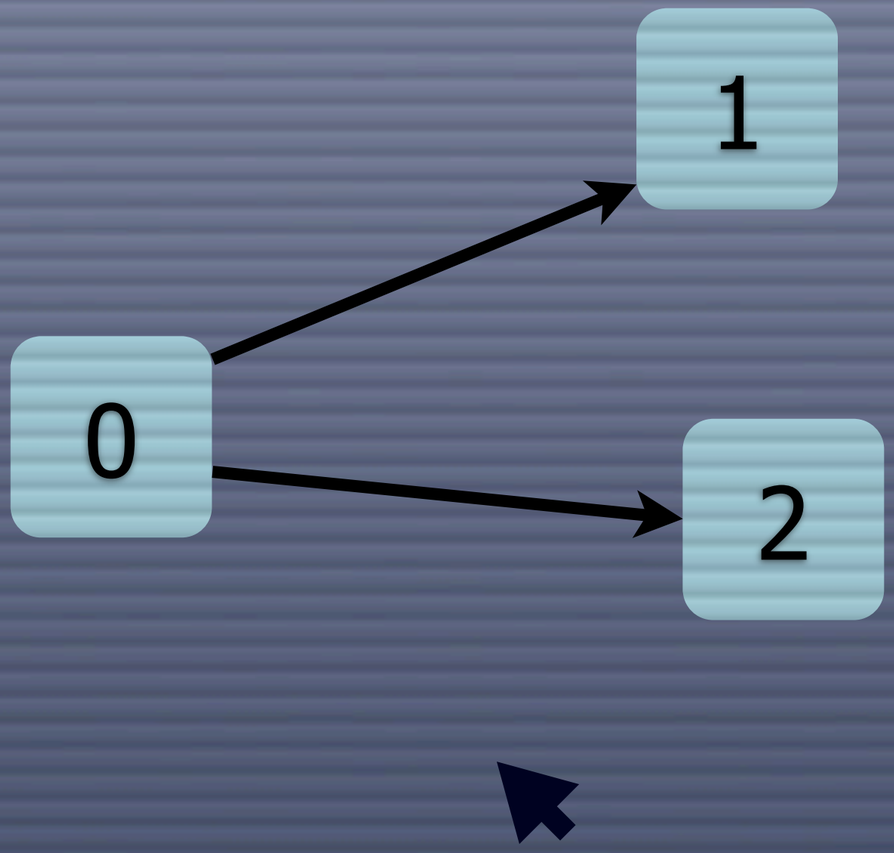
On-line Drawing of s



On-line Drawing of s

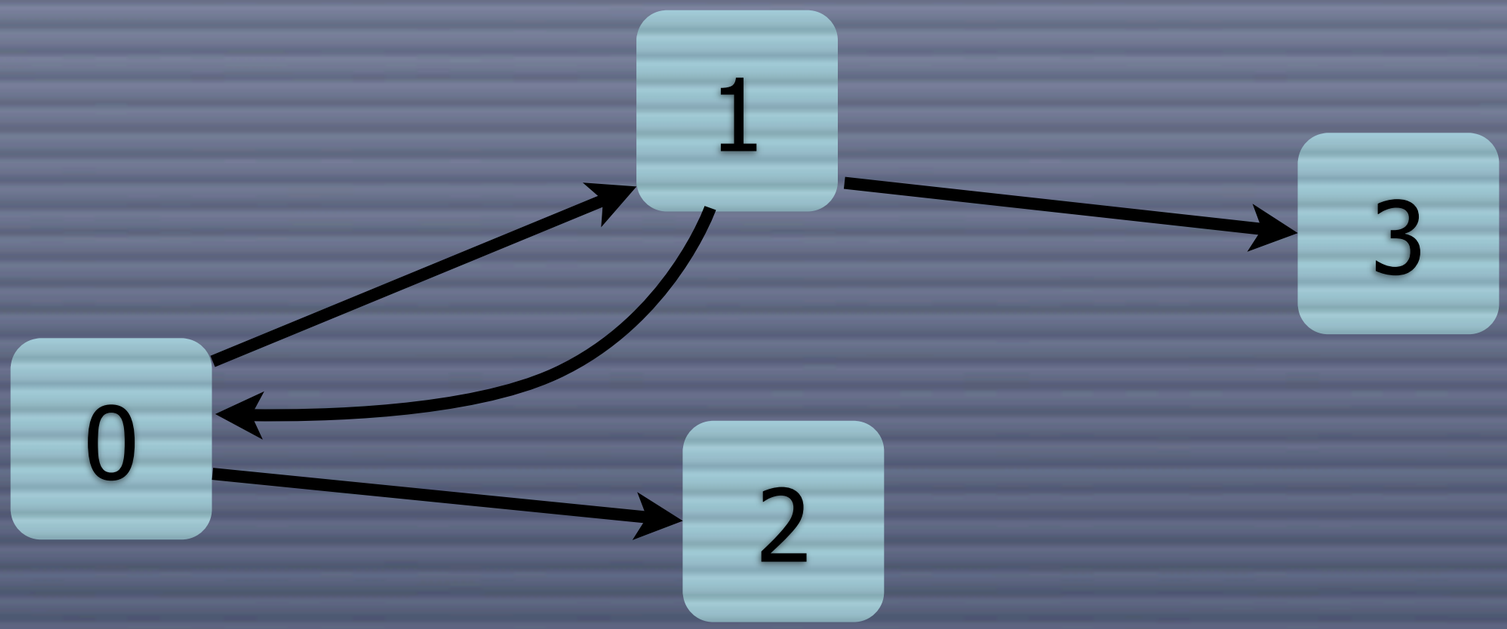


On-line Drawing of Graphs



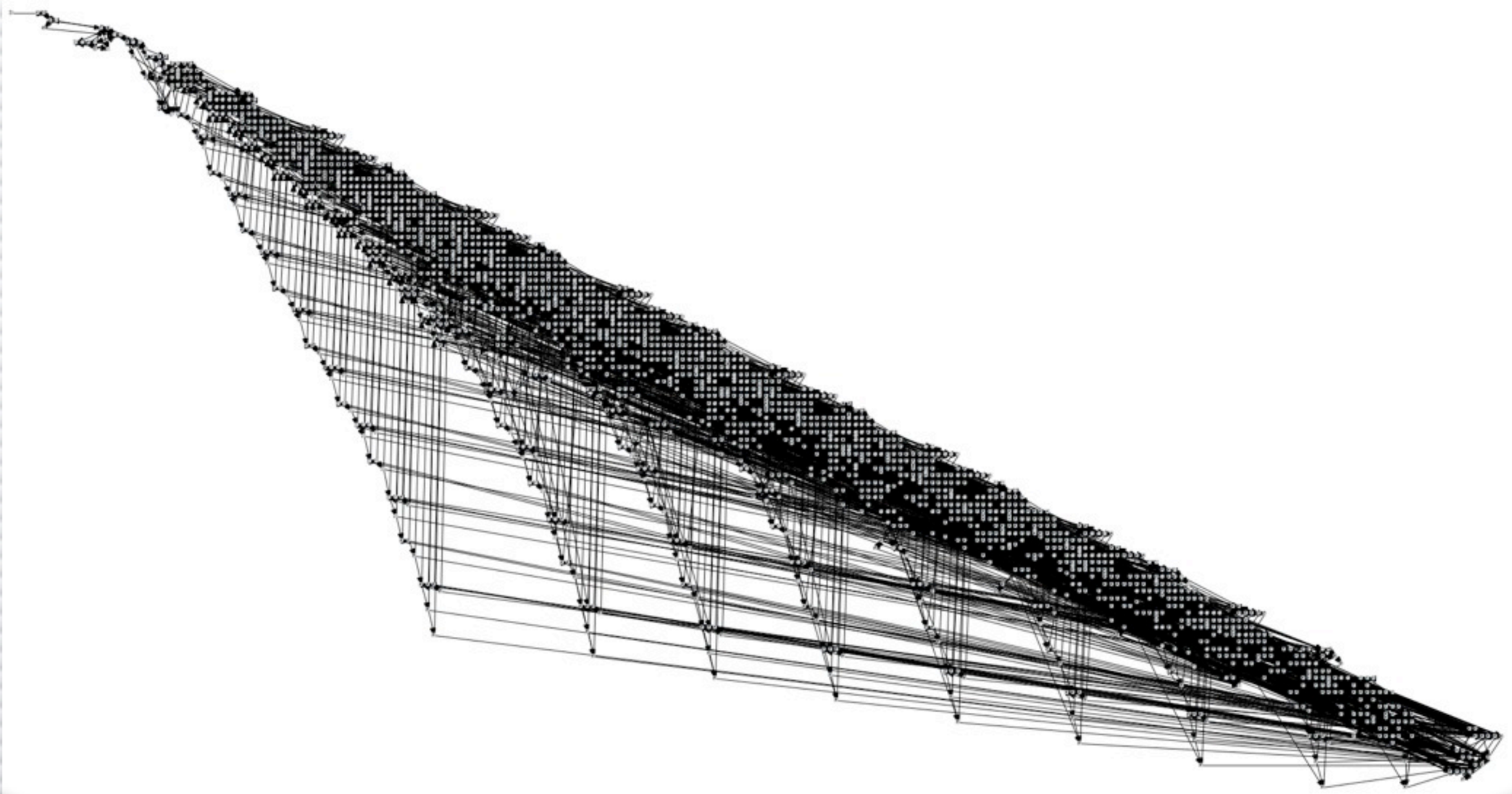
A representation of state 1 is sent to state space engine, which calculates and returns the successors

On-line Drawing of Graphs

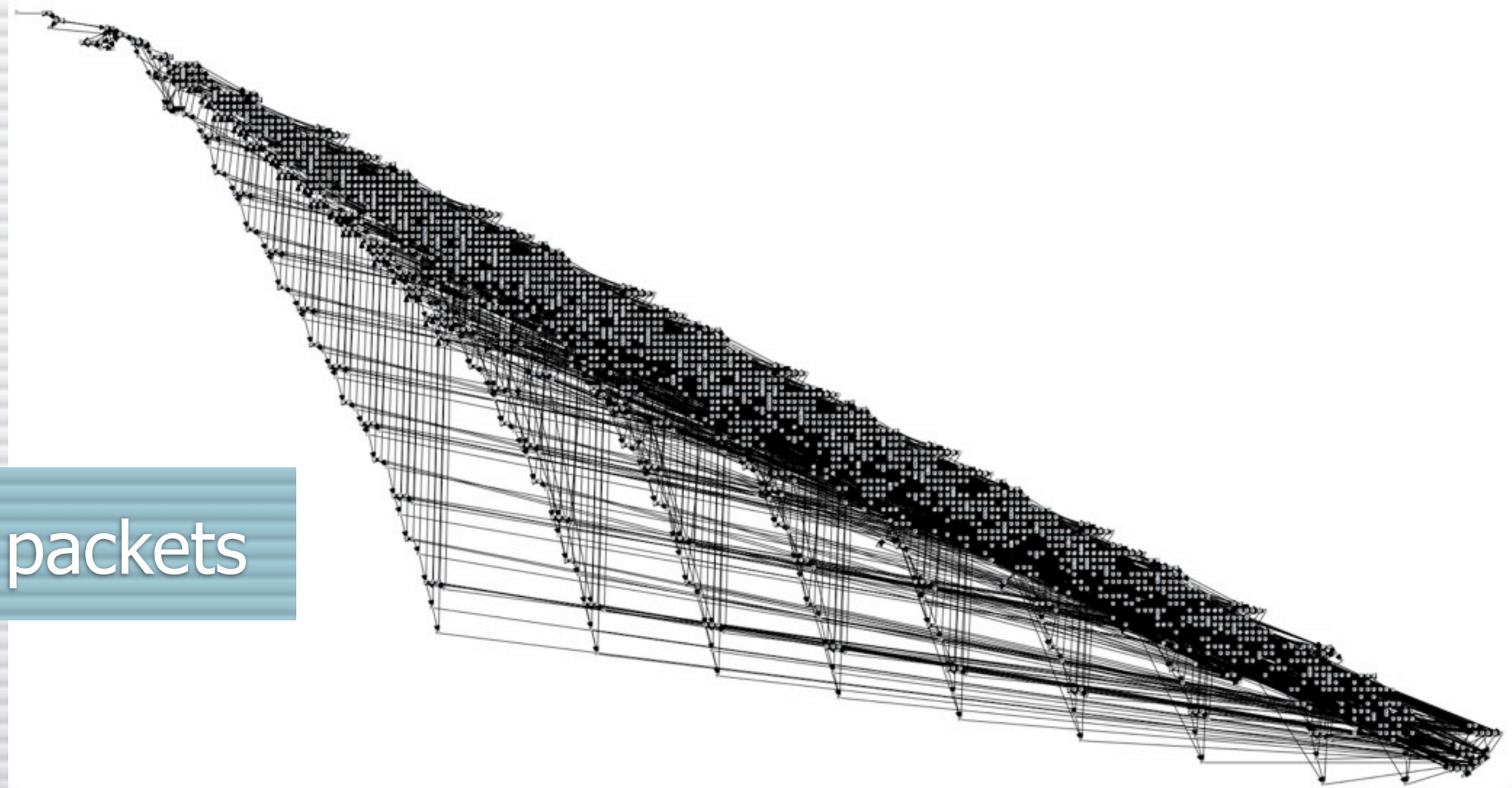


Advanced Graph Resizing using Image Resizing

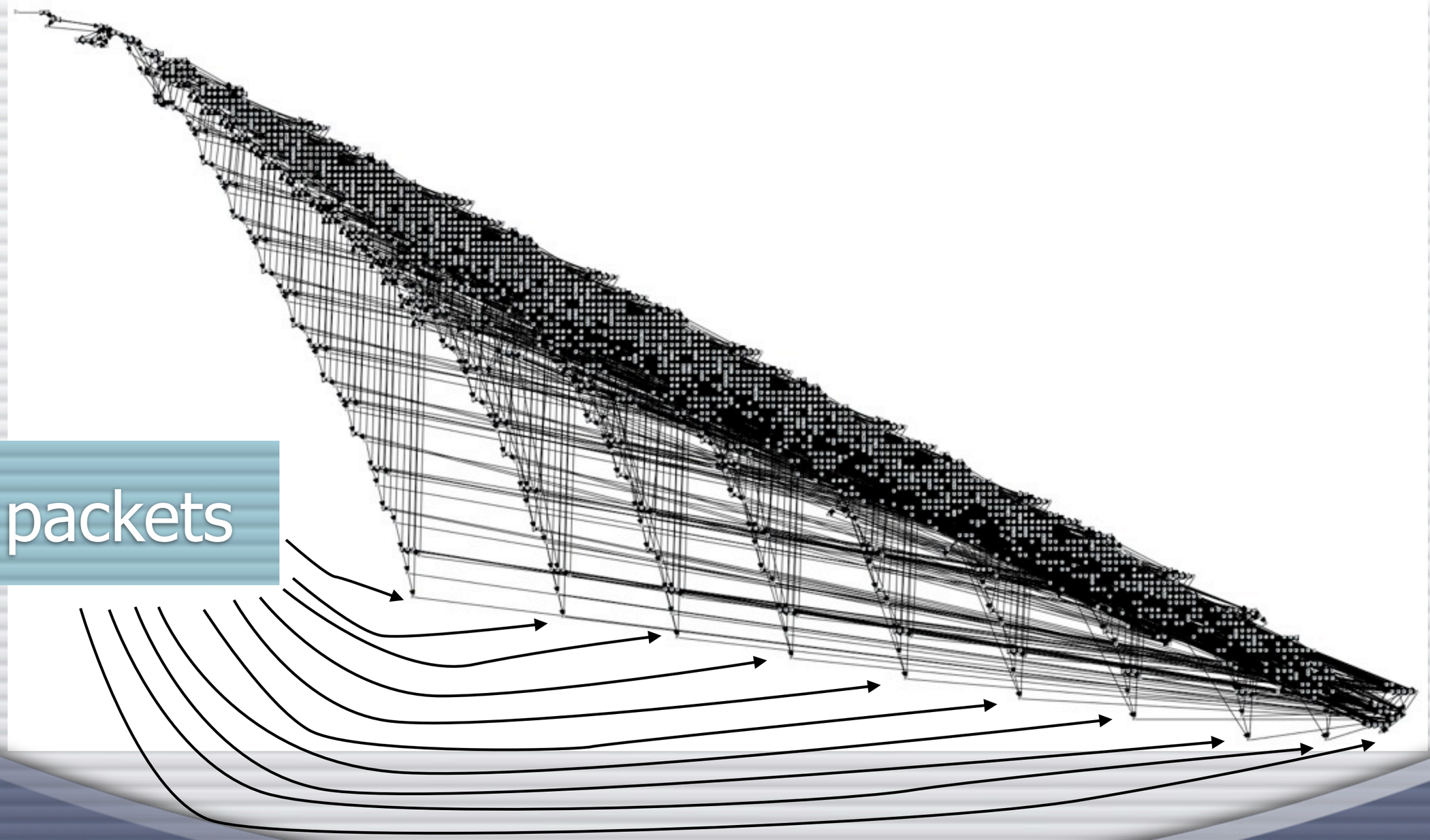
- When you initially explore the behavior of a system, you often have little idea of what to look for in the model
- A visualization of the state space may help
- For an interesting system, though, the graph is often too large to draw (in reasonable time at least)
- Master's thesis by Surayya Urazimbetova



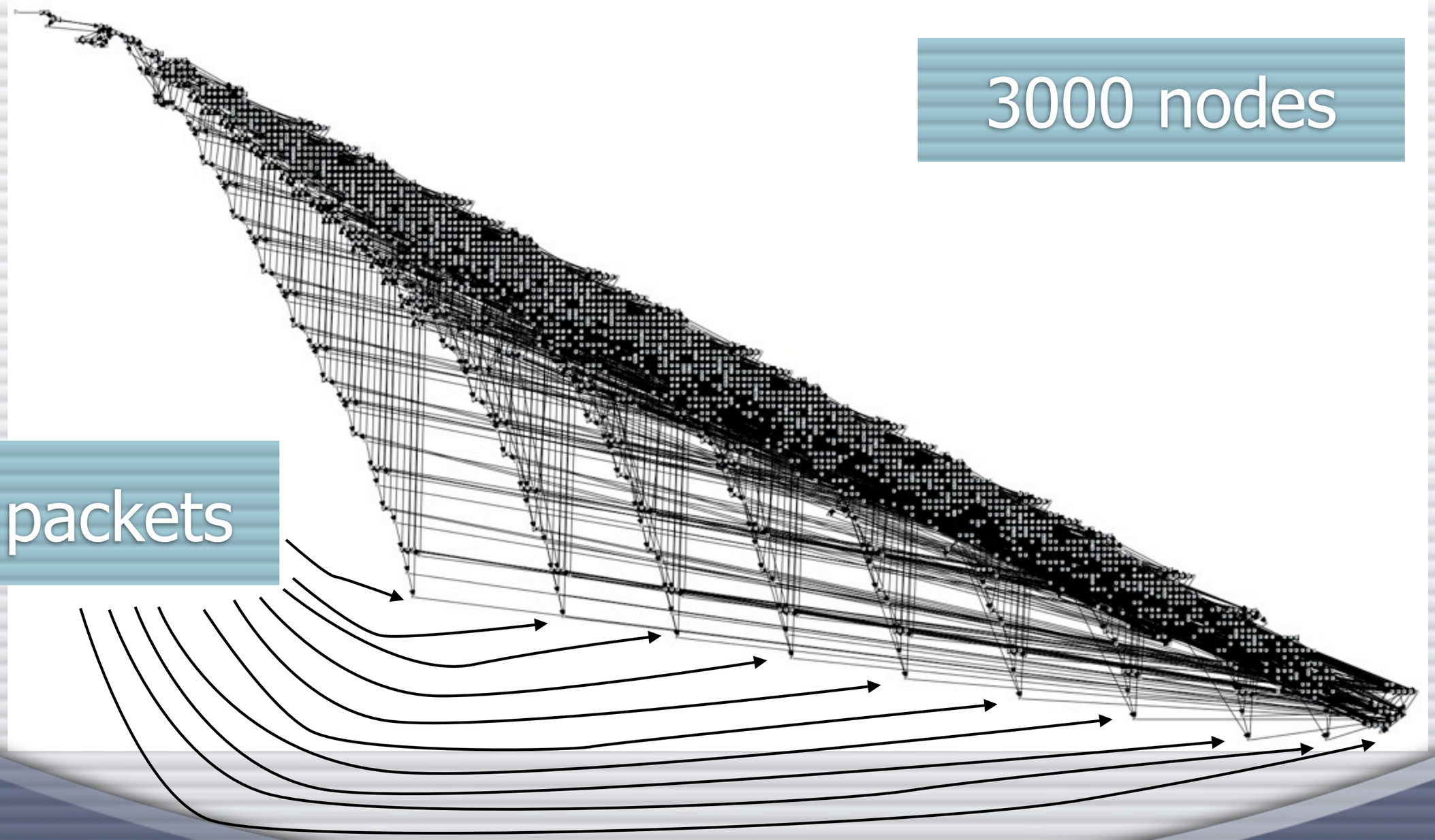
Example: Simple Protocol



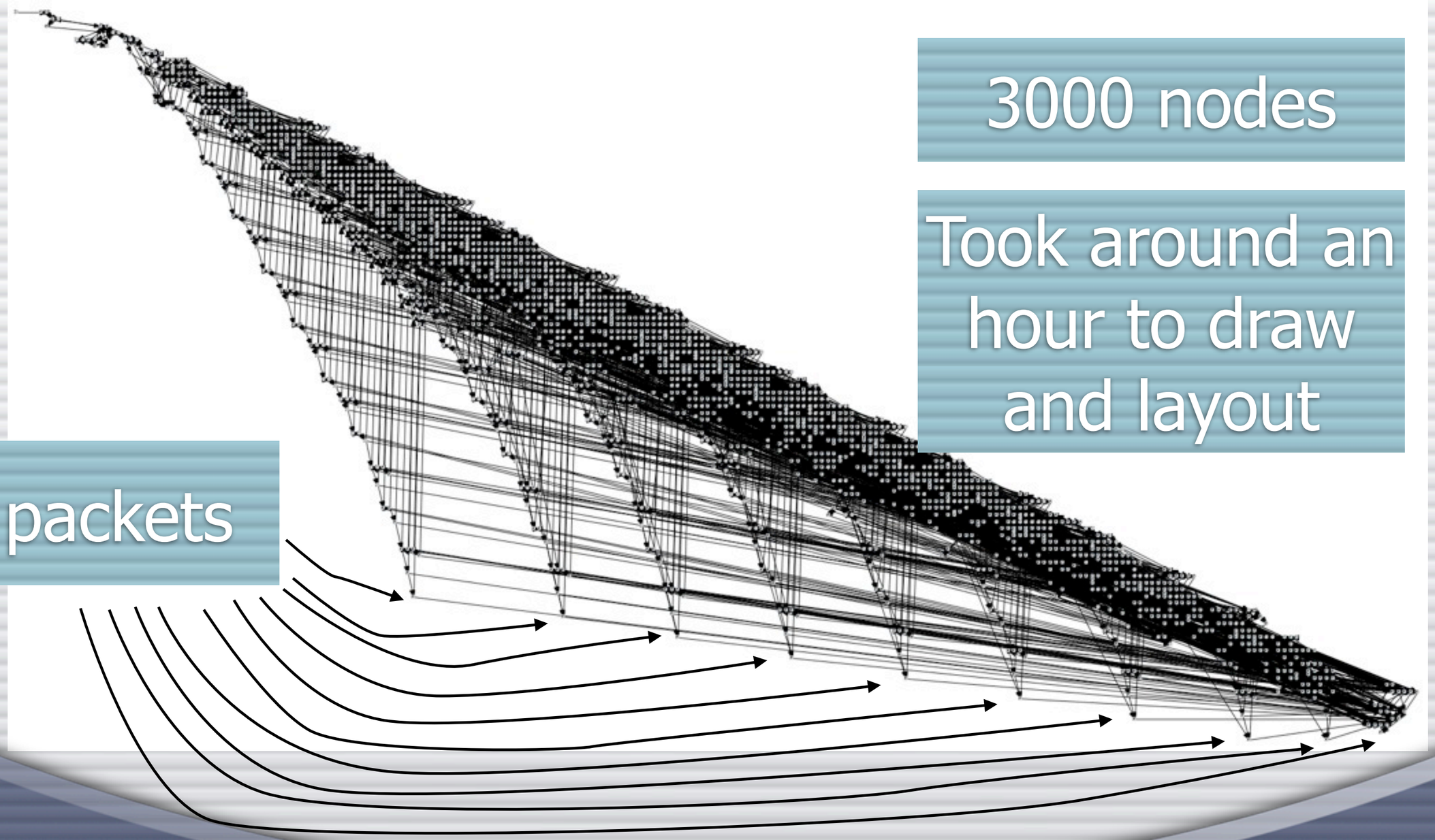
Example: Simple Protocol



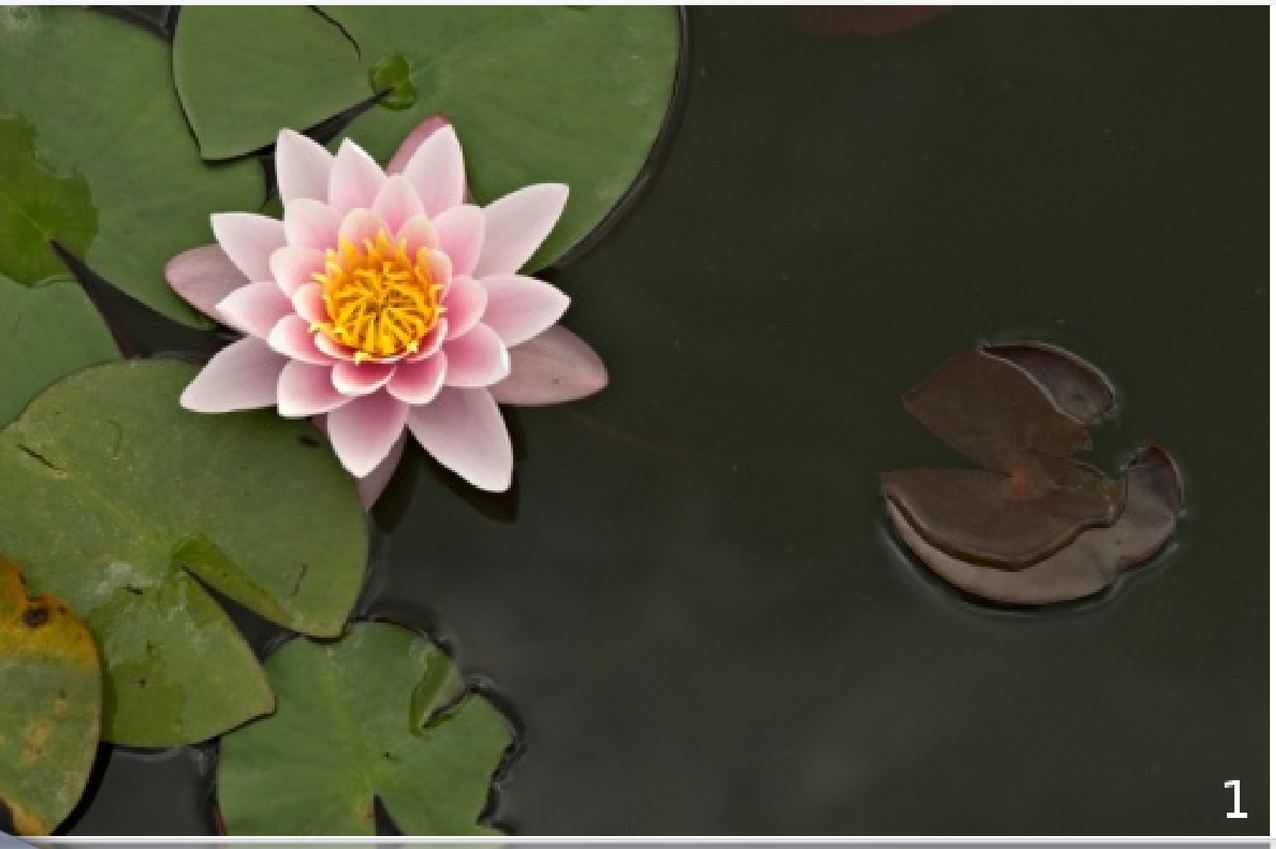
Example: Simple Protocol

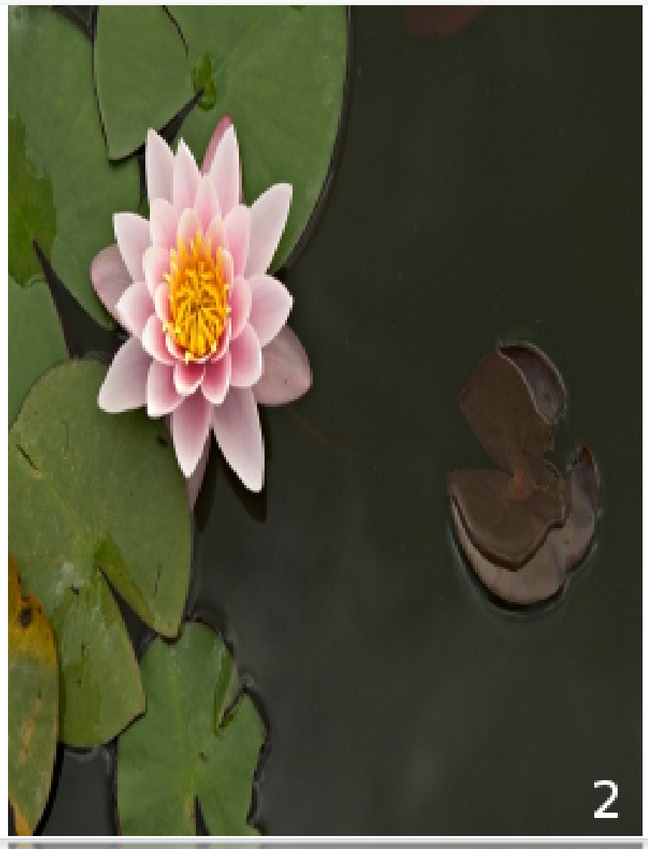


Example: Simple Protocol

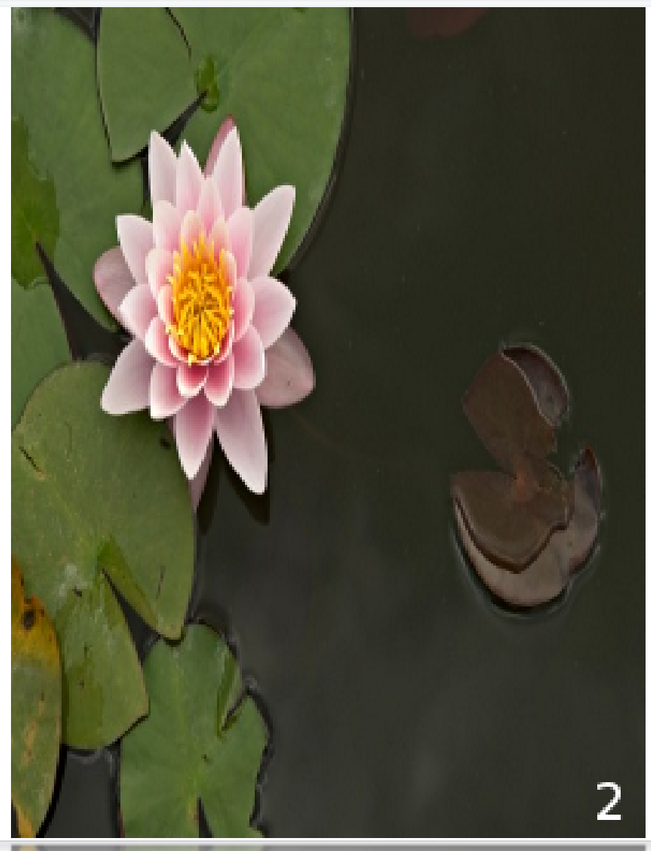


Example: Simple Protocol

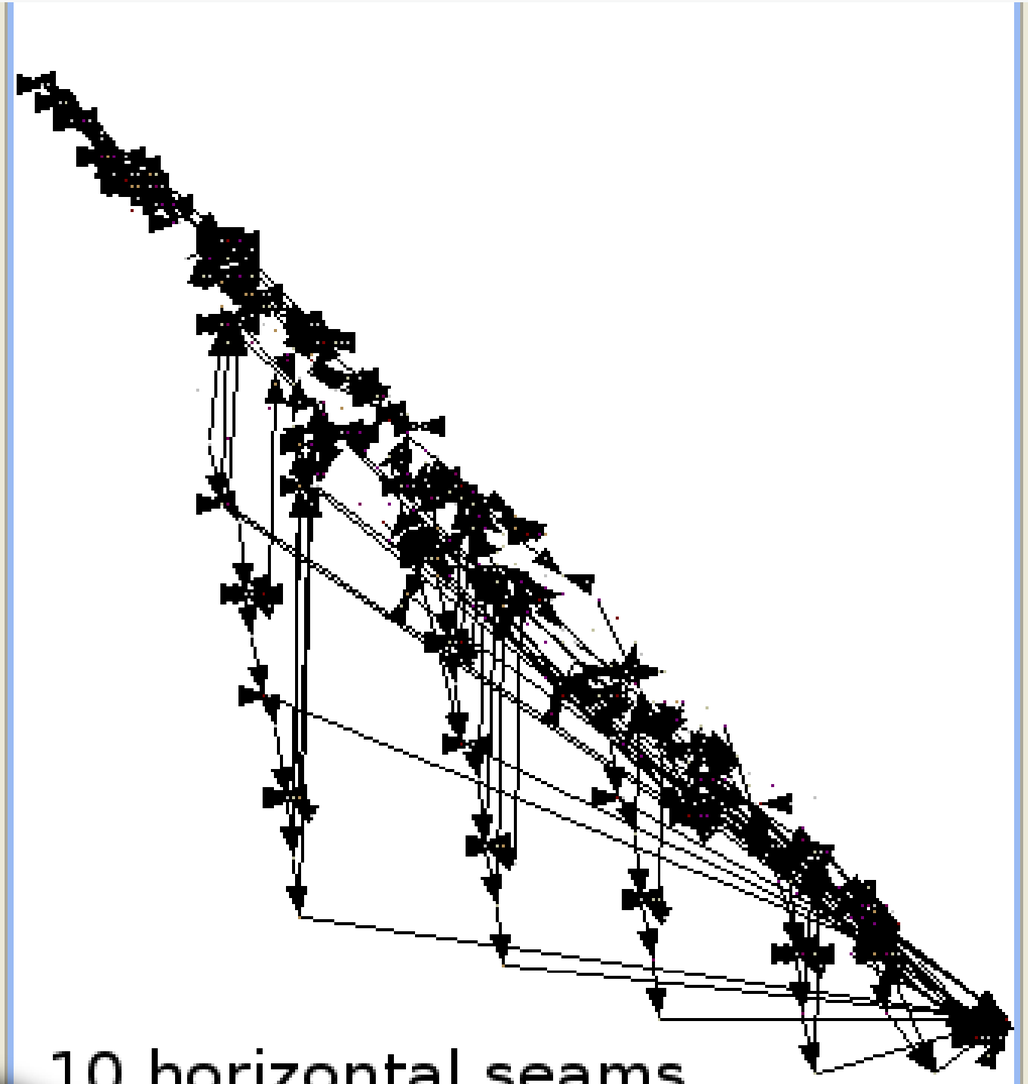




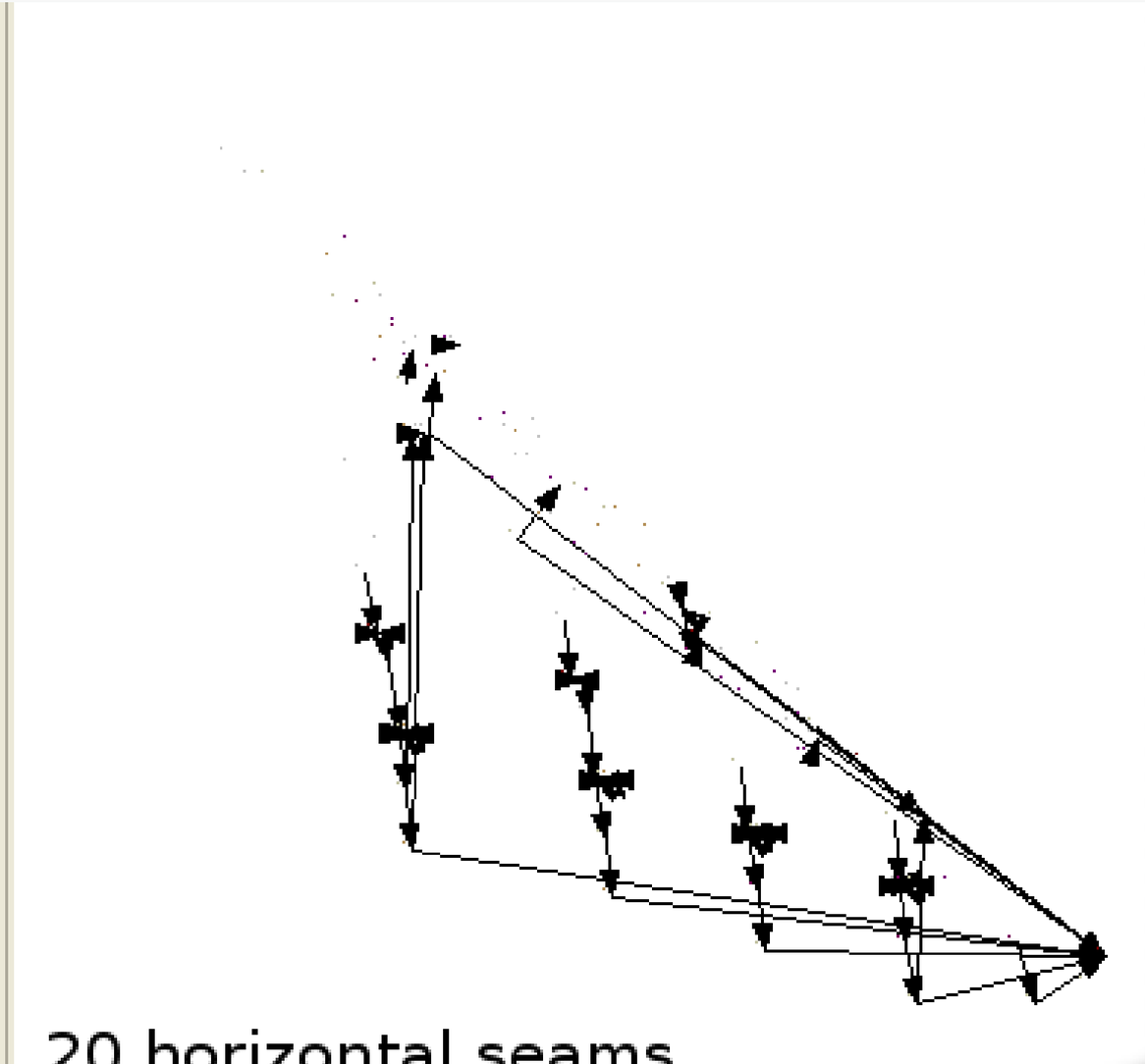
Context-Aware Image Resizing



Context-Aware Image Resizing



10 horizontal seams



20 horizontal seams

Using this for Graphs

On-the-fly LTL Model Checking

- LTL is a standard logic for specifying generalized liveness properties
- We are currently working on integrating a LTL model checker into ASAP

Longer-term New Features

- Off-line CTL analysis
- Distributed (safety) checking
- Extend JoSEL (syntactical sugar, langauge extensions)
- Integrate CPN viewer
- More user friendly ways to specify

Get It!

- ASAP can be downloaded from www.cs.au.dk/CPnets/projects/ascoveco/asap.html
- Access/CPN can be downloaded from www.cs.au.dk/CPnets/projects/ascoveco/accesscpn
- Videos from tutorial available on YouTube: tinyurl.com/asaptutorial09

Get It!

Thanks for you attention!

- ASAP can be downloaded from www.cs.au.dk/CPnets/projects/ascoveco/asap.html
- Access/CPN can be downloaded from www.cs.au.dk/CPnets/projects/ascoveco/accesscpn
- Videos from tutorial available on YouTube: tinyurl.com/asaptutorial09