Iterating transducers

Dennis Dams Yassine Lakhnech

Martin Steffen

TU/e Eindhoven and *Bell Labs, Lucent*

Verimag Grenoble

Inst. für Informatik u. Prakt. Mathematik CAU Kiel

Overview

- model checking and regular languages
- transducers
- iterating transducers
- conclusion

Infinite state model checking

- specifically nasty instance of state explosion: infinite many states
- reasons: infinite data, infinite control (e.g. parameterized systems), time ...
- scores of approaches:
  - use your own brain (and time ...): theorem proving
  - abstraction
  - symbolic techniques (many)

- 3 questions:
  1. how to represent infinite sets of states
  2. how to represent the transition relation?
  3. how to calculate the reachable states in a finite amount of time?

Regular model checking

- *very* successful finite description/symbolic representation of infinite "objects": regular languages

$\Rightarrow$ regular model checking (e.g., for parameterized systems $P_1 \parallel P_2 \parallel \ldots$, (cf. [JN00][ABJ98][KMM$^+$97] $\ldots$)

  - local states as letters of an alphabet
  - global states as linear arrangement of local ones = word

  $\Rightarrow$ infinite sets of states = reg. language

  $\Rightarrow$ computation step, i.e., non-det. change of language = transduction

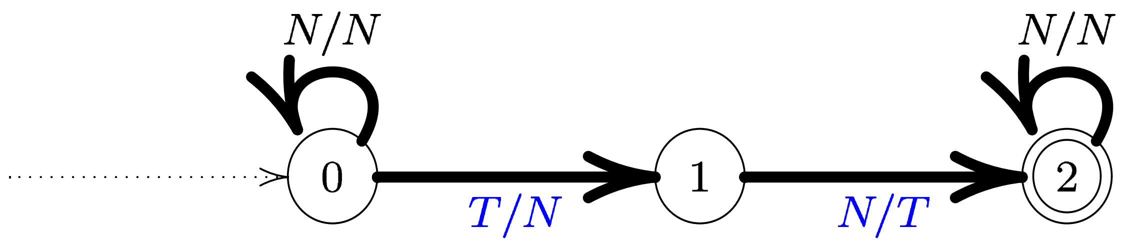
Example

Token array: “Parameterized” processes: each one either has the token or not (states T and N). Token can be passed between neighbors from left to right, initially, the token is owned by the left-most process.

Initial configuration:	TN^*
one step:	$TN \to NT$

Example (cont'd)

- one-step reduction relation: captured by a transducer



- *e.g.:* $\mathcal{T}(NTNN) = \{NNTN\}$

$\Rightarrow$ exploit for symbolic exploration: $\mathcal{T}^n \circ \mathcal{A}$

$$= \{t' \in \mathcal{T}^n(t) \mid \text{and } t \text{ accepted by } \mathcal{A}\}$$
$$= \{t' \mid t \rightarrow^n t', t \text{ accepted by } \mathcal{A}\}$$

Goal: iterating transducers

- assuming that you know how to calculate $\mathcal{T}_1 \circ \mathcal{T}_2$ by a product construction:

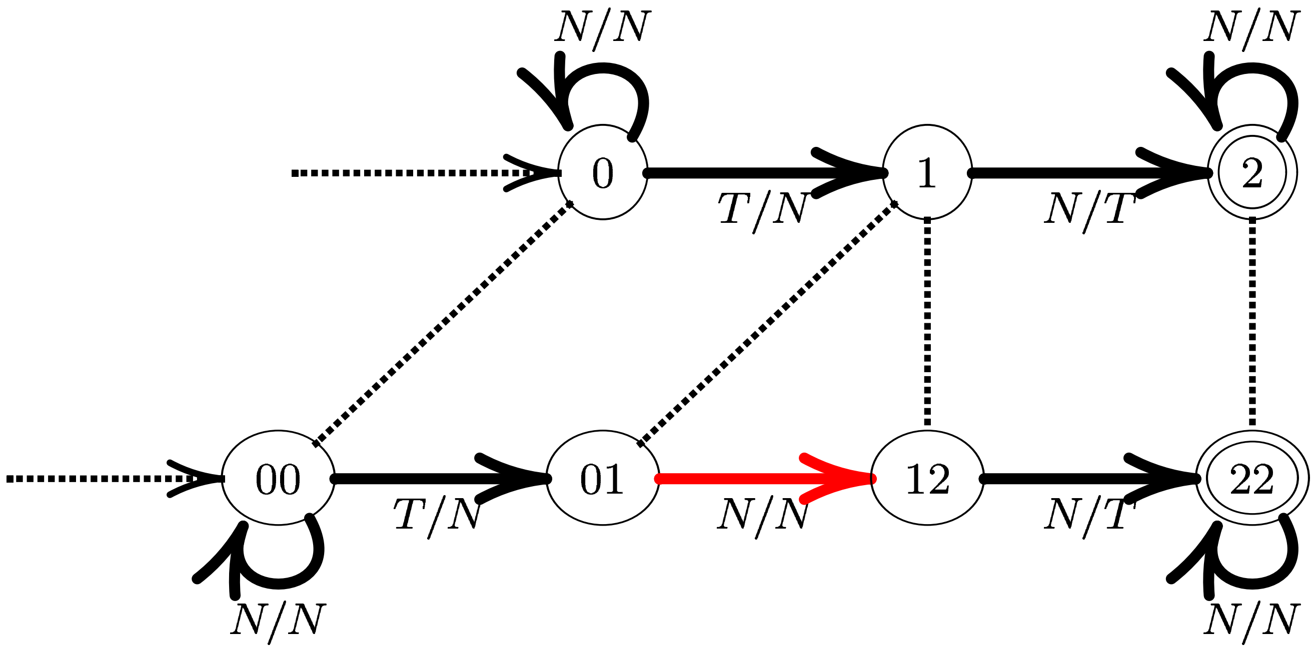
  calculate $\mathcal{T}^*$ as fixpoint $\mu X.\mathcal{T} \circ (X \cup \mathcal{T}_{id})$?

  1. $\mathcal{T}^*$ may not be representable as finite transducer (e.g.: duplicating the number of letter a: $q_0 a(x) \to aaq_0(x)$)
  2. even if: iterating naïvely $\mu X.\mathcal{T} \circ (X \cup \mathcal{T}_{id})$ will in general

$\boxed{\text{diverge}}$

tfrrr

Example: first 2 iterations



A finite representation for $\mathcal{T}^*$?

- a sound infinite representation $\mathcal{T}^{<\omega}$ for $\mathcal{T}^*$ is straightforward (using Q^* as set of states)

$\Rightarrow$ for a finite representation: build a quotient $\mathcal{T}^{<\omega}_{/\cong}$

$\Rightarrow$ remains:

1. what to take for $\cong$?
2. how to compute $\mathcal{T}^{<\omega}_{/\cong}$?

Key observation for quotienting

Theorem 1 (Soundness) *given* $F, P \subseteq Q^*$

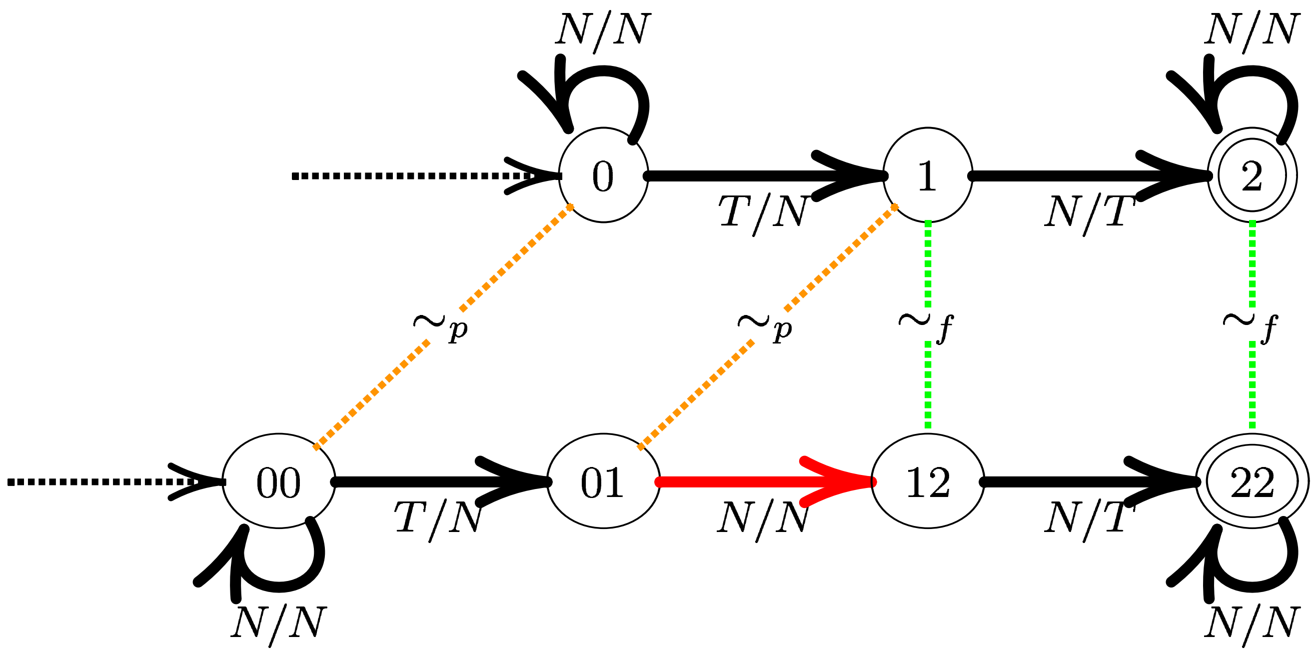
- F *and* P *two bisimulations (future and past)*
- F *and* P *swap, meaning that*

$$F; P = P; F$$

$\Rightarrow$

$$\boxed{[\![\mathcal{T}^{<\omega}]\!] = [\![\mathcal{T}^{<\omega}_{/F;P}]\!]}$$

Example, revisited



$$q_{01} \sim_p q_1 \sim_f q_{12}$$

But still: how to compute $\mathcal{T}^{<\omega}_{/F;P}$ ***?***

$\mathcal{T}^{<\omega}$ is infinite! (for Q^* is)

- way out:
  - calculate bisim's P and F on finite approximations $\mathcal{T}^{\leq n}$
  - "extrapolate" to $\mathcal{T}^{<\omega}$
- how to extrapolate?

tfrrr

Extrapolation

$\Rightarrow$ use rewriting theory, replace P and F by $\Longleftrightarrow_P^*$ and $\Longleftrightarrow_F^*$.

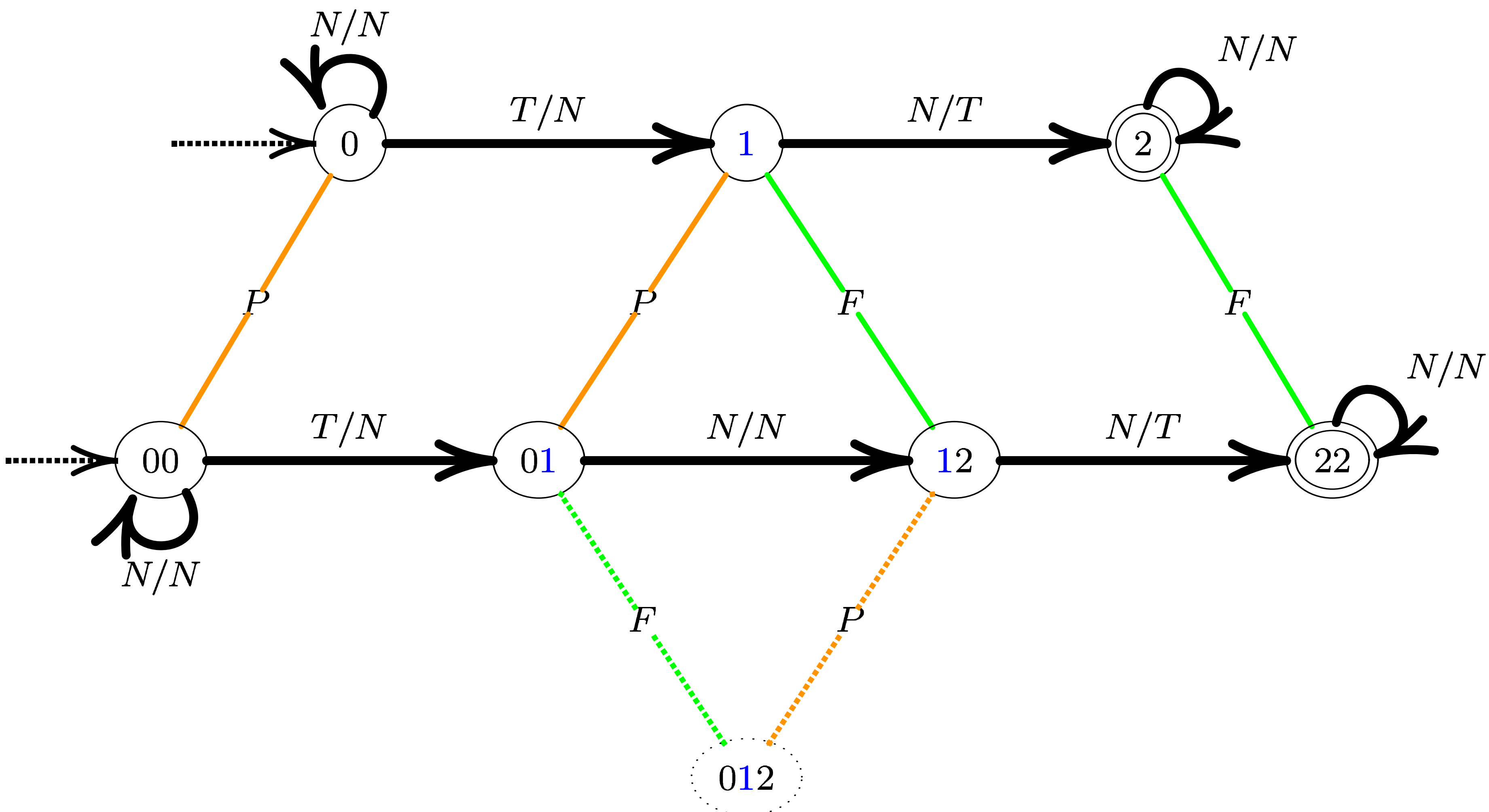
- bisimulations are congruences wrt. to the monoid Q^*
- extrapolate swapping condition (for instance):

  if $\leftrightarrow_P$ and $\leftrightarrow_F$ are confluent and swap, then so are $\Longleftrightarrow_P^*$ and $\Longleftrightarrow_F^*$

$\Rightarrow$ bisimulations found in finite $\mathcal{T}^{\leq n}$ can be used to quotient $\mathcal{T}^{<\omega}$

tfrrr

Example



Algorithm

input $\mathcal{T} = (Q, Q_i, Q_f, \Sigma, R)$
$\mathcal{X} := \mathcal{T}_{id}$;
repeat
$\mathcal{X} := (\mathcal{T} \circ \mathcal{X}) \cup \mathcal{T}_{id}$;
determine bisimulations F and P on $\mathcal{X}$ s.t.
$\leftrightarrow_F$ and $\leftrightarrow_P$ swap and each possess the diamond property;
until $\mathcal{X}_{/\equiv} \sim_f (\mathcal{T} \circ \mathcal{X}_{/\equiv}) \cup \mathcal{T}_{id}$

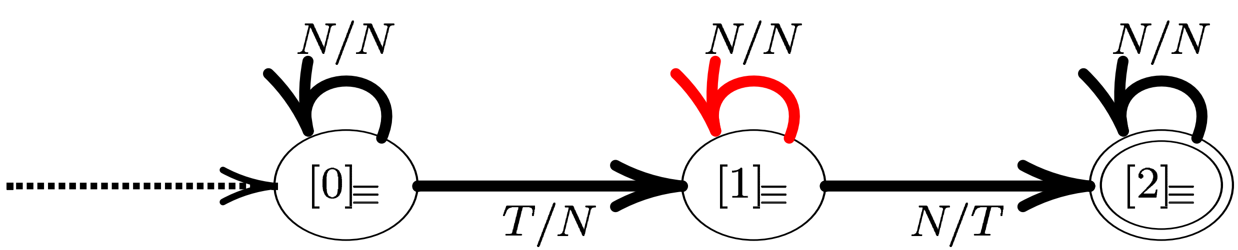
Example

- Rewrite system after 2 iterations:

i.e.

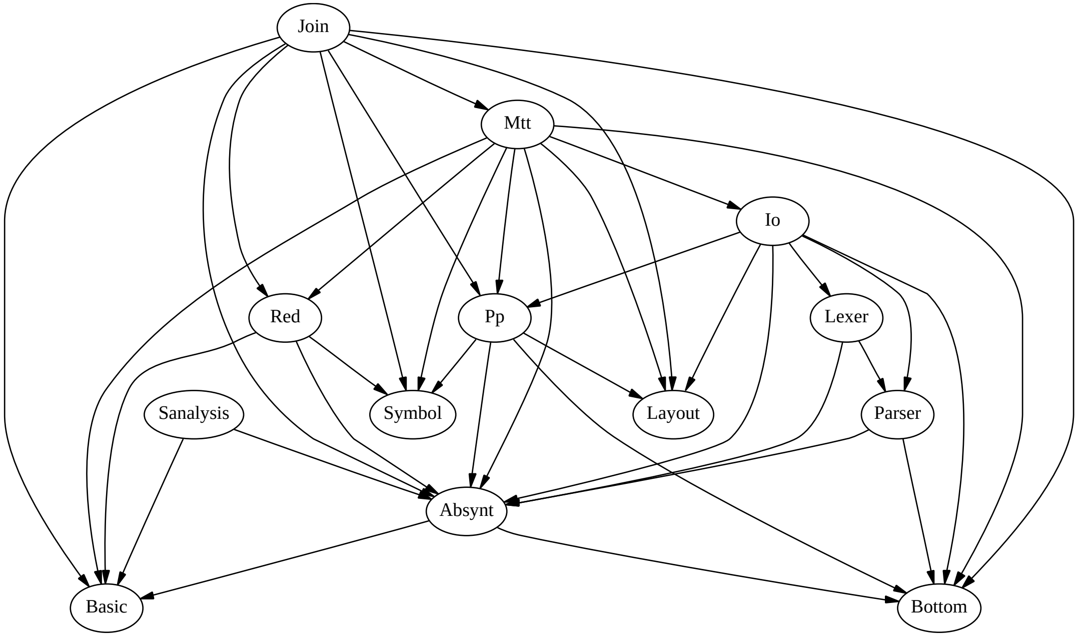
$$\begin{array}{lcl} 00 & \rightarrow & 0 \\ 01 & \rightarrow & 1 \\ 12 & \rightarrow & 1 \\ 22 & \rightarrow & 2 \end{array}$$

$$\begin{aligned} [\![0]\!] &= \{0, 00, \ldots\}, \\ [\![1]\!] &= \{1, 01, 001, \ldots, 12, 122, \ldots\}, \\ [\![2]\!] &= \{2, 22, \ldots\}. \end{aligned}$$



Implementation

- library of transducer-operations (iteration, composition, transduction)
- in *ocaml*
- efficiency: sufficient for small examples



Conclusion

- characterize iterateable transducers, complexity?
- ϵ-transitions and weak bisimulation?
- Compare with
  - monadic string rewriting [BO93]
  - column-transducers of k-bounded depth [Nil00]
- specialize to: $\mathcal{T}^{\leq n} \circ \mathcal{A}$. benefits?
- more complicated examples, dynamic process creation
- implementation: efficiency, various optimizations
- further into the jungle of tree transducers ...

References

[ABJ98] Parosh Aziz Abdulla, Ahmed Bouajjani, and Bengt Jonsson. On-the-fly analysis of systems with unbounded lossy Fifo-channels. In Alan J. Hu and Moshe Y. Vardi, editors, *Proceedings of CAV '98*, volume 1427 of *Lecture Notes in Computer Science*, pages 305–318. Springer-Verlag, 1998.

[BO93] Ronald Book and Friedrich Otto. *String Rewriting Systems*. Monographs in Computer Science. Springer-Verlag, 1993.

[JN00] Bengt Jonsson and Marcus Nilsson. Transitive closures of regular relations for verifying infinite-state systems. In S. Graf and M. Schwartzbach, editors, *TACAS 2000*, volume 1785 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.

[KMM+97] Y. Kesten, O. Maler, M. Marcus, A. Pnueli, and E. Shahar. Symbolic model checking with rich assertional languages. In Orna Grumberg, editor, *CAV '97, Proceedings of the 9th International Conference on Computer-Aided Verification, Haifa. Israel*, volume 1254 of *Lecture Notes in Computer Science*. Springer, June 1997.

[Nil00] Marcus Nilsson. Regular model checking, 2000. Licenciate Thesis of Uppsala University, Department of Information Technology.