Weakly-Synchronized Ground Tree Rewriting

(with Applications to Verifying Multithreaded Programs)

Anthony Widjaja Lin

Oxford University Department of Computer Science

Introduction

Multithreaded programs

- Rapidly gaining popularity
- Multithreading supported by JAVA, Python, C++, C#
- Benefit: concurrent executions on multiple processors
- **Main problem**:
  - Can be difficult to understand
  - Standard testing and debugging insufficient

parbegin/parend in action:

```
int* mergesort(int *array)
  int *a1, *a2;
  parbegin
     a1 = mergesort(1st half of array);
     a2 = mergesort(2nd half of array);
  parend
  return merge(a1,a2);
```

Parallel.For in action:

```
bool b[50];

Parallel.For(0,49,i,=>)
{
  b[i] = fun2(a[i]);
}

return ⋀_{i=0}^{49} b[i];
```

Summary of the problem

- **Problem**: verify multithreaded programs with `parbegin`/`parend` and `parallel.for` constructs.
- **Our approach**:
  - Design a formal model
  - Design verification algorithms

The model

Summary of the model

- **Pushdown systems (PDS)**: popular model for sequential programs with function calls.

PDS $\subseteq$ GTRS $\subseteq$ wGTRS $\subseteq$ sGTRS

Summary of the model

- **Pushdown systems (PDS)**: popular model for sequential programs with function calls.
- **Ground Tree Rewrite Systems (GTRS)**: extends PDS & captures `Parallel.For` and `parbegin/parend` constructs w/ no shared variables.

$$\text{PDS} \subseteq \text{GTRS} \subseteq \text{wGTRS} \subseteq \text{sGTRS}$$

Summary of the model

- **Pushdown systems (PDS)**: popular model for sequential programs with function calls.
- **Ground Tree Rewrite Systems (GTRS)**: extends PDS & captures `Parallel.For` and `parbegin`/`parend` constructs w/ no shared variables.
- **State-extended GTRS (sGTRS)**: extends GTRS by allowing shared variables. (Turing-complete!)

PDS $\subseteq$ GTRS $\subseteq$ wGTRS $\subseteq$ sGTRS

Summary of the model

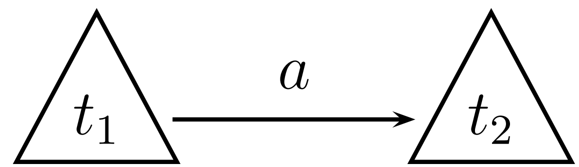
- **Pushdown systems (PDS)**: popular model for sequential programs with function calls.
- **Ground Tree Rewrite Systems (GTRS)**: extends PDS & captures `Parallel.For` and `parbegin`/`parend` constructs w/ no shared variables.
- **State-extended GTRS (sGTRS)**: extends GTRS by allowing shared variables. (Turing-complete!)
- **Weakly-Synchronized GTRS (wGTRS)**: a good decidable approximation of sGTRS.

$$\text{PDS} \subseteq \text{GTRS} \subseteq \text{wGTRS} \subseteq \text{sGTRS}$$

Ground Tree Rewrite Systems (GTRS)

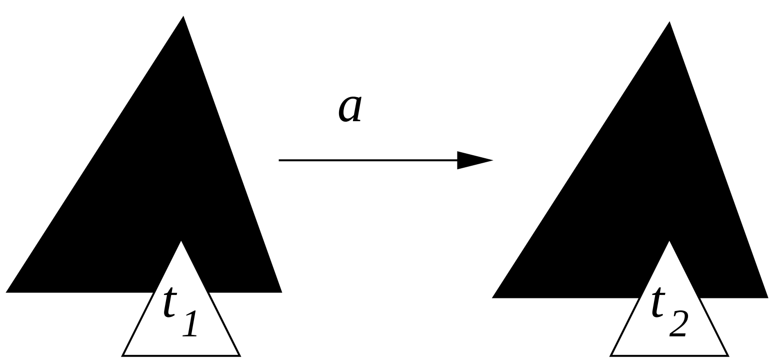
Syntax:

Consists of a finite set of "rewrite" rules that look like



Semantics as a transition system:

Domain: set of all trees (over suitable ranked alphabet)
Transitions:



GTRS example: `mergesort`

GTRS rules for arrays of four ints:

- Dividing tasks, e.g.,

$$\mathrm{msort}([1,5,10,3]) \quad \Longrightarrow \quad \begin{tikzcd} & \mathrm{msort}([1,5,10,3]) \arrow[d] \arrow[dr] & \\ & \mathrm{msort}([1,5]) & \mathrm{msort}([10,3]) \end{tikzcd}$$

- Merging, e.g.,

$$\begin{tikzcd} \mathrm{msort}([1,5,10,3]) \arrow[d] \arrow[dr] & \\ {[1,5]} & {[3,10]} \end{tikzcd} \quad \Longrightarrow \quad [1,3,5,10]$$

GTRS example: `mergesort`

msort([7,1,3,2])

GTRS example: `mergesort`

msort([7,1,3,2])

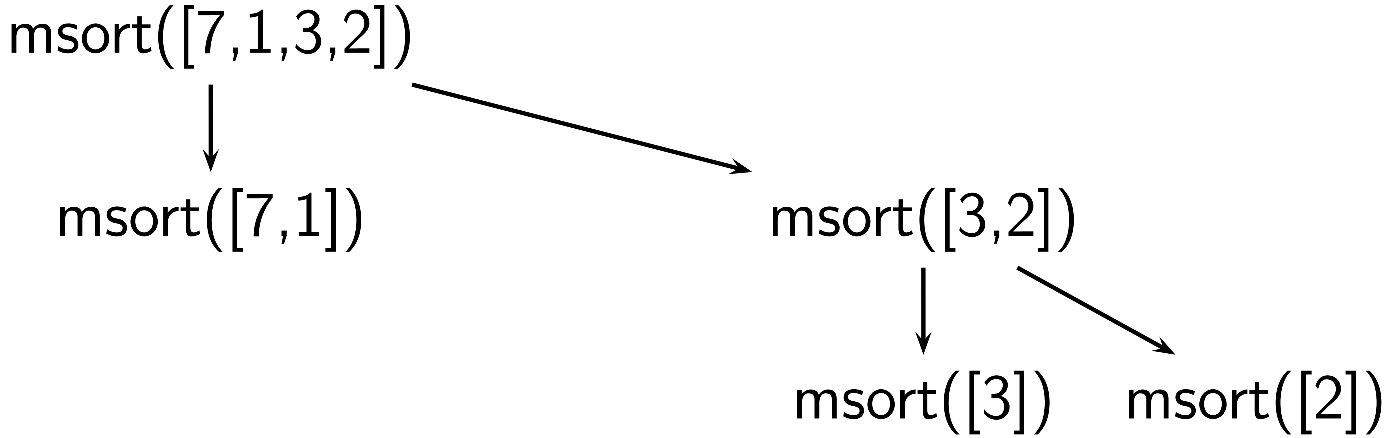
GTRS example: `mergesort`

$$\begin{tikzcd} \mathrm{msort}([7,1,3,2]) \arrow[d] \arrow[dr] & \\ \mathrm{msort}([7,1]) & \mathrm{msort}([3,2]) \end{tikzcd}$$

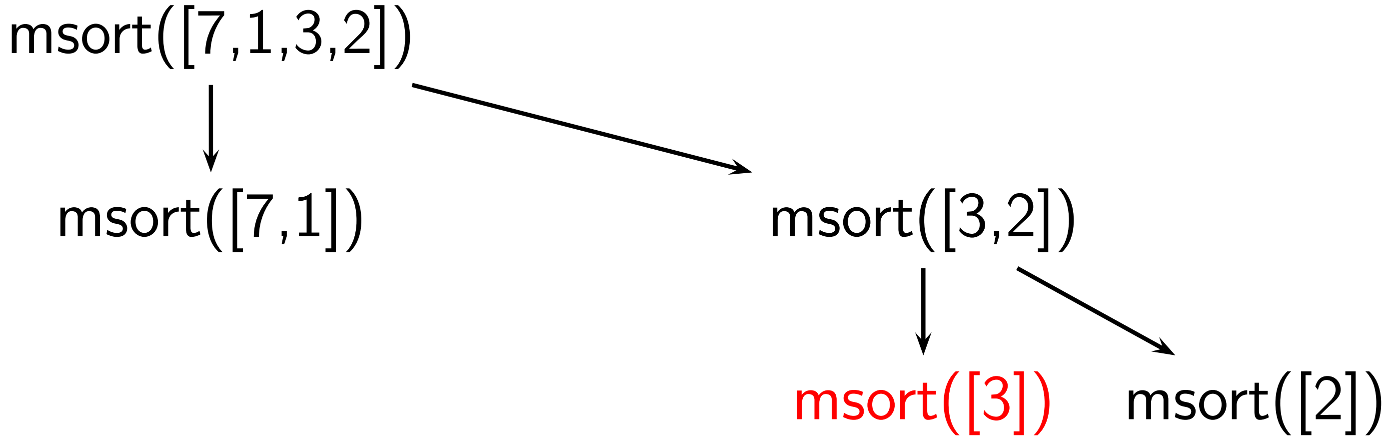
GTRS example: `mergesort`

$$
\begin{tikzcd}
\text{msort}([7,1,3,2]) \arrow[d] \arrow[dr] & \\
\text{msort}([7,1]) & \text{msort}([3,2])
\end{tikzcd}
$$

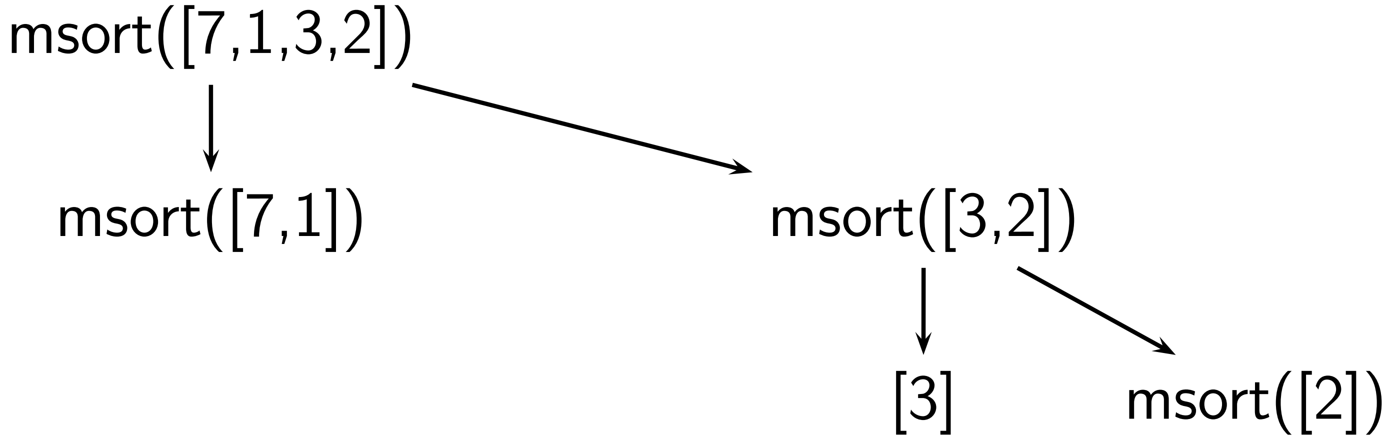
GTRS example: `mergesort`



GTRS example: `mergesort`



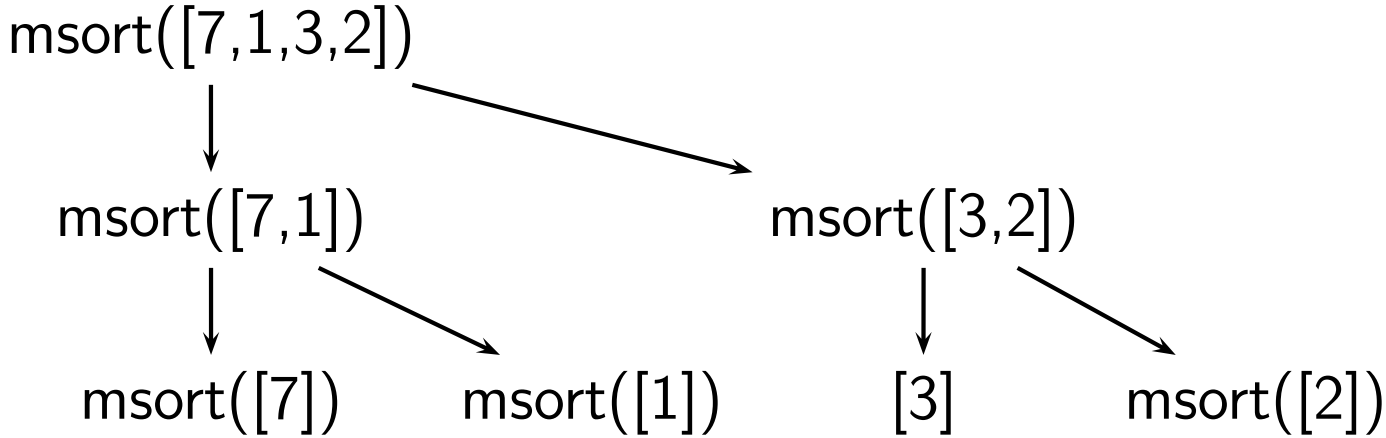
GTRS example: `mergesort`



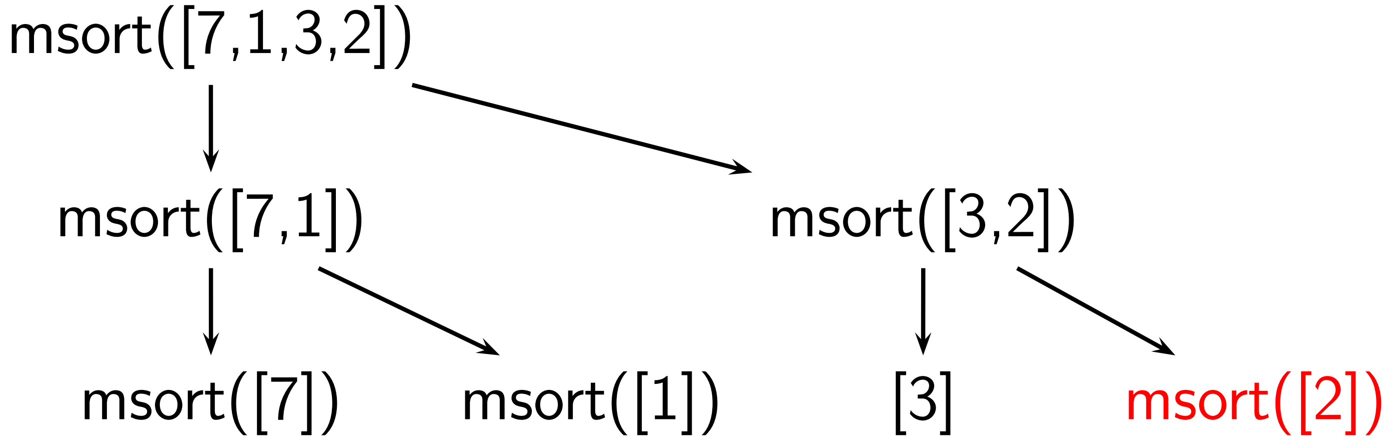
GTRS example: `mergesort`

$$
\begin{tikzcd}
\text{msort}([7,1,3,2]) \arrow[d] \arrow[drr] & & & \\
\text{msort}([7,1]) & & \text{msort}([3,2]) \arrow[d] \arrow[dr] & \\
& & [3] & \text{msort}([2])
\end{tikzcd}
$$

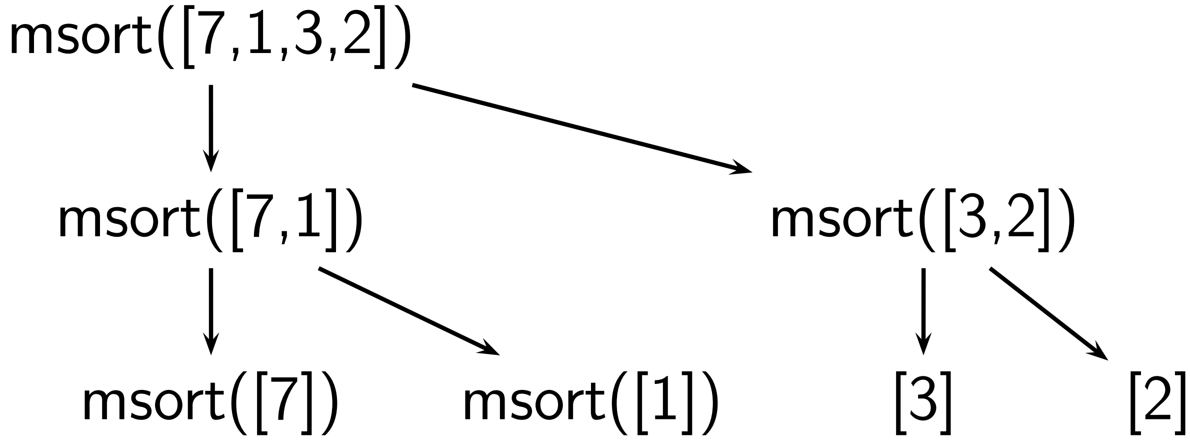
GTRS example: `mergesort`



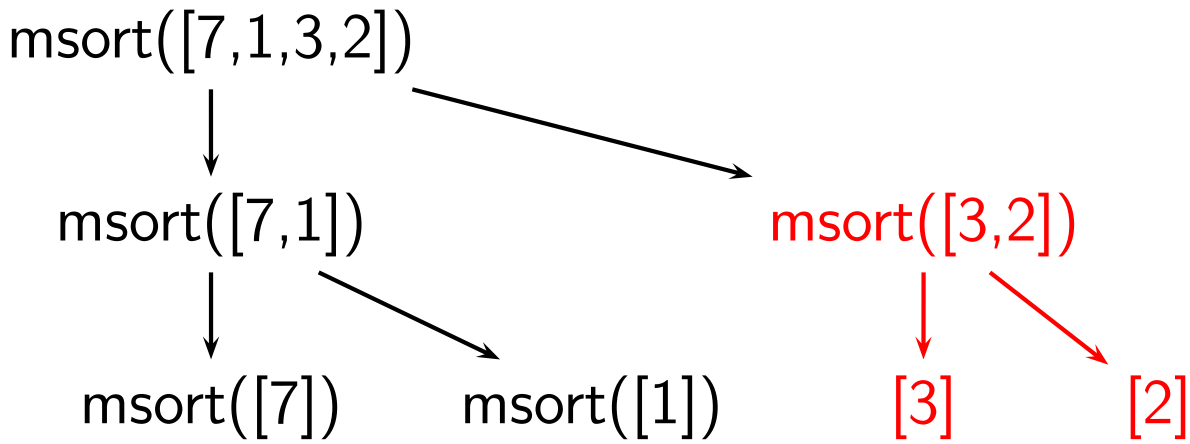
GTRS example: `mergesort`



GTRS example: `mergesort`



GTRS example: `mergesort`



GTRS example: `mergesort`

$$
\begin{tikzcd}
\text{msort}([7,1,3,2]) \arrow[d] \arrow[rr] & & {[2,3]} \\
\text{msort}([7,1]) \arrow[d] \arrow[dr] & & \\
\text{msort}([7]) & \text{msort}([1]) &
\end{tikzcd}
$$

GTRS example: `mergesort`

$$\begin{tikzcd}[column sep=small]
\text{msort}([7,1,3,2]) \arrow[d] \arrow[rr] & & {[2,3]} \\
\text{msort}([7,1]) \arrow[d] \arrow[dr] & & \\
\text{msort}([7]) & \text{msort}([1]) &
\end{tikzcd}$$

AND SO ON UNTIL ...

GTRS example: `mergesort`

$[1,2,3,7]$

State-extended GTRS (sGTRS)

- Threads often communicate via shared variables
  - **e.g.**: `count++` on calling `mergesort`
- GTRS framework cannot capture this
- In general, need to extend GTRS with states

State-extended GTRS (sGTRS)

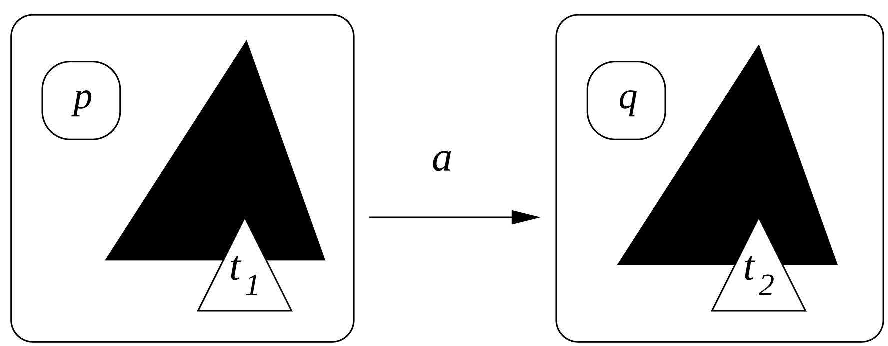
Syntax:

Rewrite rules have control state components

$$\left(p, t_1\right) \xrightarrow{a} \left(q, t_2\right)$$

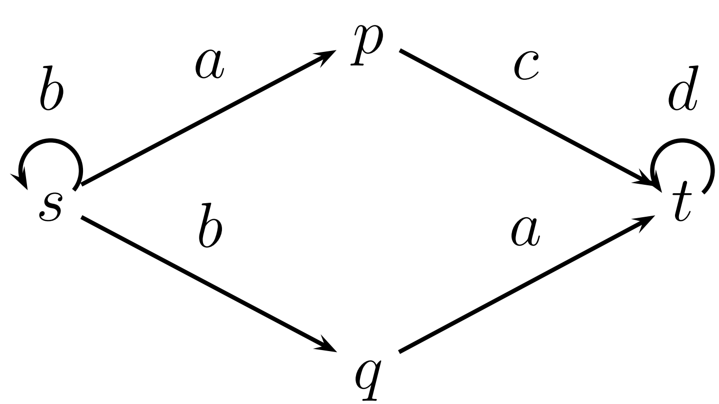
Semantics as a transition system:

Domain: {control states} $\times$ {all trees}
Transitions:



Weakly-Synchronized GTRS (wGTRS)

- sGTRS can simulate 2-stack automata (Turing-comp!)
- For decidability, restrict the underlying control graph:
  - omit tree component of rewrite rules
- **wGTRS**: restrict to DAG with self-loops (a.k.a. weak control unit)



Weakly-Synchronized GTRS (wGTRS)

What good wGTRS for?

- Timing and event constraints among threads
- Captures sGTRS runs up to bounded # of syncs (many concurrency bugs occur within ≤ 5 syncs)

Our results

Statements of main results

Theorem: Reachability for wGTRS is NP-complete. Moreover, it can be efficiently reduced to existential Presburger theory.

- Highly optimised solvers for existential Presburger theory are available (e.g. Z3).
- Corollaries:
  - Repeated reachability is NP-complete.
  - Model checking a fragment of LTL is coNP-complete.

Reducing to existential Presburger theory

wGTRS Reachability

Instance: $\underbrace{\mathcal{G}}_{\text{wGTRS}}, \underbrace{\left(p, t_1\right)}_{\text{start conf.}}, \underbrace{\left(q, t_2\right)}_{\text{final conf.}}$

Question: $\left(p, t_1\right) \longrightarrow^* \left(q, t_2\right)$?

Reducing to existential Presburger theory

Overview of the reduction:

- ☐ Construct CFG “simulating” wGTRS (more behavior)

Reducing to existential Presburger theory

Overview of the reduction:

- Construct CFG "simulating" wGTRS (more behavior)
- Restrict CFG behavior to derivation trees satisfying linear arithmetic constraints ψ on # occurrences of terminals (Parikh image)

Reducing to existential Presburger theory

Overview of the reduction:

- □ Construct CFG “simulating” wGTRS (more behavior)
- □ Restrict CFG behavior to derivation trees satisfying linear arithmetic constraints ψ on $\#$ occurrences of terminals (Parikh image)
- □ Use PTIME algo from *Verma et al.'06* computing Parikh image of CFG as exist. Presburger formula φ

Reducing to existential Presburger theory

Overview of the reduction:

- □ Construct CFG "simulating" wGTRS (more behavior)
- □ Restrict CFG behavior to derivation trees satisfying linear arithmetic constraints ψ on # occurrences of terminals (Parikh image)
- □ Use PTIME algo from *Verma et al.'06* computing Parikh image of CFG as exist. Presburger formula φ
- □ wGTRS reachability instance is positive iff $\langle \mathbb{N}; + \rangle \models \varphi \wedge \psi$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{c} X \\ \diagup \quad \diagdown \\ X \qquad X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \qquad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{c} & X & \\ \diagup & & \diagdown \\ X & & X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, {\color{red}X})$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \qquad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{c} X \\ \diagup \quad \diagdown \\ X \qquad X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \rightarrow \left(q, \begin{array}{c} X \\ \diagup \quad \diagdown \\ \color{red}{X} \qquad X \end{array}\right)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{c} X \\ / \quad \backslash \\ X \qquad X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{array}{c} X \\ / \quad \backslash \\ X \qquad X \end{array}\right) \to \left(q, \begin{array}{c} X \\ / \quad \backslash \\ Y \qquad X \end{array}\right)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \qquad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \rightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right) \rightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & {\color{red}X} \end{array}\right)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \qquad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right) \to \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & X \end{array}\right) \to \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q,\ \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \rightarrow \left(q,\ \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right) \rightarrow \left(q,\ \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & X \end{array}\right) \rightarrow \left(q,\ \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)$$

The corresponding CFG derivation:

$$N_{(p,X),\left(q,\ \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)}$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \rightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right) \rightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & X \end{array}\right) \rightarrow \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)$$

The corresponding CFG derivation:

$$N_{\color{blue}{(p,X)}, \color{red}{\left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)}}$$

$$\rightarrow N_{\color{blue}{(p,X)}, \left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right)} \quad N_{\left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ X & & X \end{array}\right), \color{red}{\left(q, \begin{array}{ccc} & X & \\ / & & \backslash \\ Y & & Y \end{array}\right)}}$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right) \to \left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & X \end{smallmatrix}\right) \to \left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)$$

The corresponding CFG derivation:

$$\begin{aligned} & N_{(p,X),\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)} \\ & \qquad \to N_{(p,X),\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right)} \quad N_{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right),\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)} \\ & \qquad \to T_{p,q} \quad N_{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right),\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)} \end{aligned}$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{smallmatrix} & X & \\ X & & X \end{smallmatrix}\right), \quad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{smallmatrix} & X & \\ X & & X \end{smallmatrix}\right) \to \left(q, \begin{smallmatrix} & X & \\ Y & & X \end{smallmatrix}\right) \to \left(q, \begin{smallmatrix} & X & \\ Y & & Y \end{smallmatrix}\right)$$

The corresponding CFG derivation:

$$\begin{aligned}
& N_{(p,X),\left(q, \begin{smallmatrix} & X & \\ Y & & Y \end{smallmatrix}\right)} \\
& \qquad \to N_{(p,X),\left(q, \begin{smallmatrix} & X & \\ X & & X \end{smallmatrix}\right)} \quad N_{\left(q, \begin{smallmatrix} & X & \\ X & & X \end{smallmatrix}\right),\left(q, \begin{smallmatrix} & X & \\ Y & & Y \end{smallmatrix}\right)} \\
& \qquad \to T_{p,q} \quad N_{\left(q, \begin{smallmatrix} & X & \\ X & & X \end{smallmatrix}\right),\left(q, \begin{smallmatrix} & X & \\ Y & & Y \end{smallmatrix}\right)} \\
& \qquad \to T_{p,q} \quad N_{(q,X),(q,Y)} \quad N_{(q,X),(q,Y)}
\end{aligned}$$

Simulation of wGTRS by CFG

$$(p, X) \longrightarrow \left(q, \begin{array}{ccc} & X & \\ \diagup & & \diagdown \\ X & & X \end{array}\right), \qquad (q, X) \longrightarrow (q, Y)$$

A run of this wGTRS:

$$(p, X) \to \left(q, \begin{array}{ccc} & X & \\ \diagup & & \diagdown \\ X & & X \end{array}\right) \to \left(q, \begin{array}{ccc} & X & \\ \diagup & & \diagdown \\ Y & & X \end{array}\right) \to \left(q, \begin{array}{ccc} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{array}\right)$$

The corresponding CFG derivation:

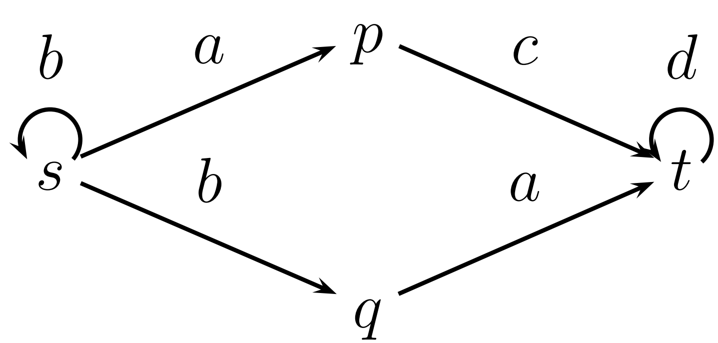
$$\begin{aligned}
& N_{\color{blue}{(p,X)}, \color{red}{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)}} \\
& \quad \to N_{\color{blue}{(p,X)}, \left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right)} \quad N_{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right), \color{red}{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)}} \\
& \quad \to T_{p,q} \quad N_{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ X & & X \end{smallmatrix}\right), \color{red}{\left(q, \begin{smallmatrix} & X & \\ \diagup & & \diagdown \\ Y & & Y \end{smallmatrix}\right)}} \\
& \quad \to T_{p,q} \quad N_{(q,X),(q,Y)} \quad N_{(q,X),(q,Y)} \\
& \quad \to T_{p,q} \quad T_{q,q} \quad T_{q,q}
\end{aligned}$$

Restricting CFG behavior via Presburger constraints

Required Constraint: Parikh image of word generated by CFG corresponds to a subgraph of the control graph of wGTRS, whose shape is chain possibly with self-loops.

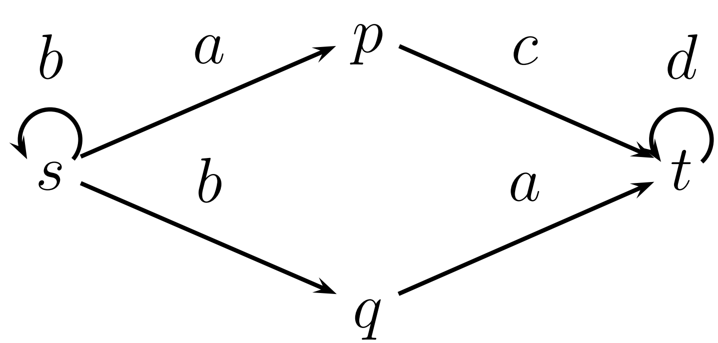
Restricting CFG behavior via Presburger constraints

Required Constraint: Parikh image of word generated by CFG corresponds to a subgraph of the control graph of wGTRS, whose shape is chain possibly with self-loops.



Restricting CFG behavior via Presburger constraints

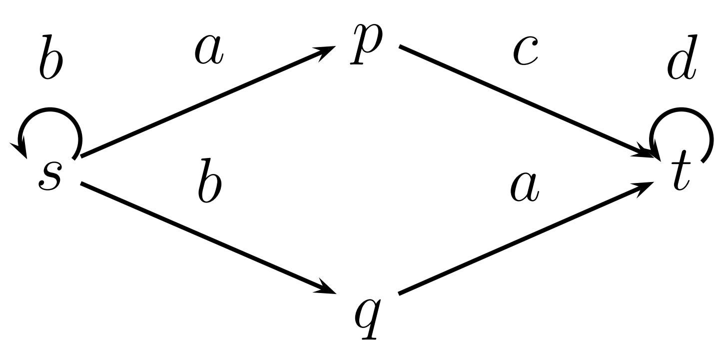
Required Constraint: Parikh image of word generated by CFG corresponds to a subgraph of the control graph of wGTRS, whose shape is chain possibly with self-loops.



$T_{s,p}T_{p,t}T_{t,t}$ is good

Restricting CFG behavior via Presburger constraints

Required Constraint: Parikh image of word generated by CFG corresponds to a subgraph of the control graph of wGTRS, whose shape is chain possibly with self-loops.

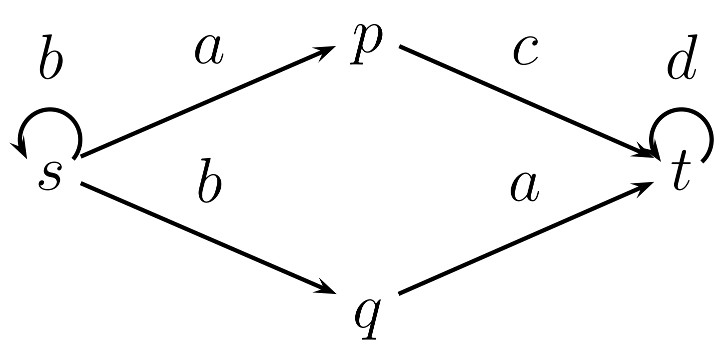


$T_{s,p}T_{p,t}T_{t,t}$ is good

$T_{s,p}T_{s,q}$ is bad

Restricting CFG behavior via Presburger constraints

Required Constraint: Parikh image of word generated by CFG corresponds to a subgraph of the control graph of wGTRS, whose shape is chain possibly with self-loops.



$T_{s,p}T_{p,t}T_{t,t}$ is good

$T_{s,p}T_{s,q}$ is bad

Required constraint is a conjunction of linear arithmetic expressions, e.g., $\#T_{s,p} > 0 \rightarrow \#T_{s,q} = 0$

Conclusion

Conclusion

- wGTRS provides good compromise between decidability and modelling power
- wGTRS can be verified by fast reduction to existential Presburger theory (and hence NP or coNP complete)

Conclusion

- wGTRS provides good compromise between decidability and modelling power
- wGTRS can be verified by fast reduction to existential Presburger theory (and hence NP or coNP complete)

THANKS FOR LISTENING!