Solving Q-SAT in bounded space and time by geometrical computation

Denys Duchier[1], Jérôme Durand-Lose[2], Maxime Senot[2]

[1] Team *Constraint & Machine Learning*
[2] Team *Graphs & Algorithms*

Laboratoire d'Informatique Fondamentale d'Orléans,
Université d'Orléans, Orléans, FRANCE

Partially supported by the ANR AGAPE, ANR-09-BLAN-0159-03.

CiE '11, Sofia, Bulgaria — 28th June 2011

Q-SAT

Decision problem Q-SAT

Input : a quantified boolean formula ϕ.
Question : Is ϕ true or false?

Example

$$\phi = \exists x_1 \forall x_2 \forall x_3 \quad x_1 \wedge (\neg x_2 \vee x_3)$$

Theorem [Stockmeyer,1973]

Q-SAT is **PSPACE**-complete.

On our classical model of computation
with the usual notion of complexity.

Brute force solution

Recursive algorithm

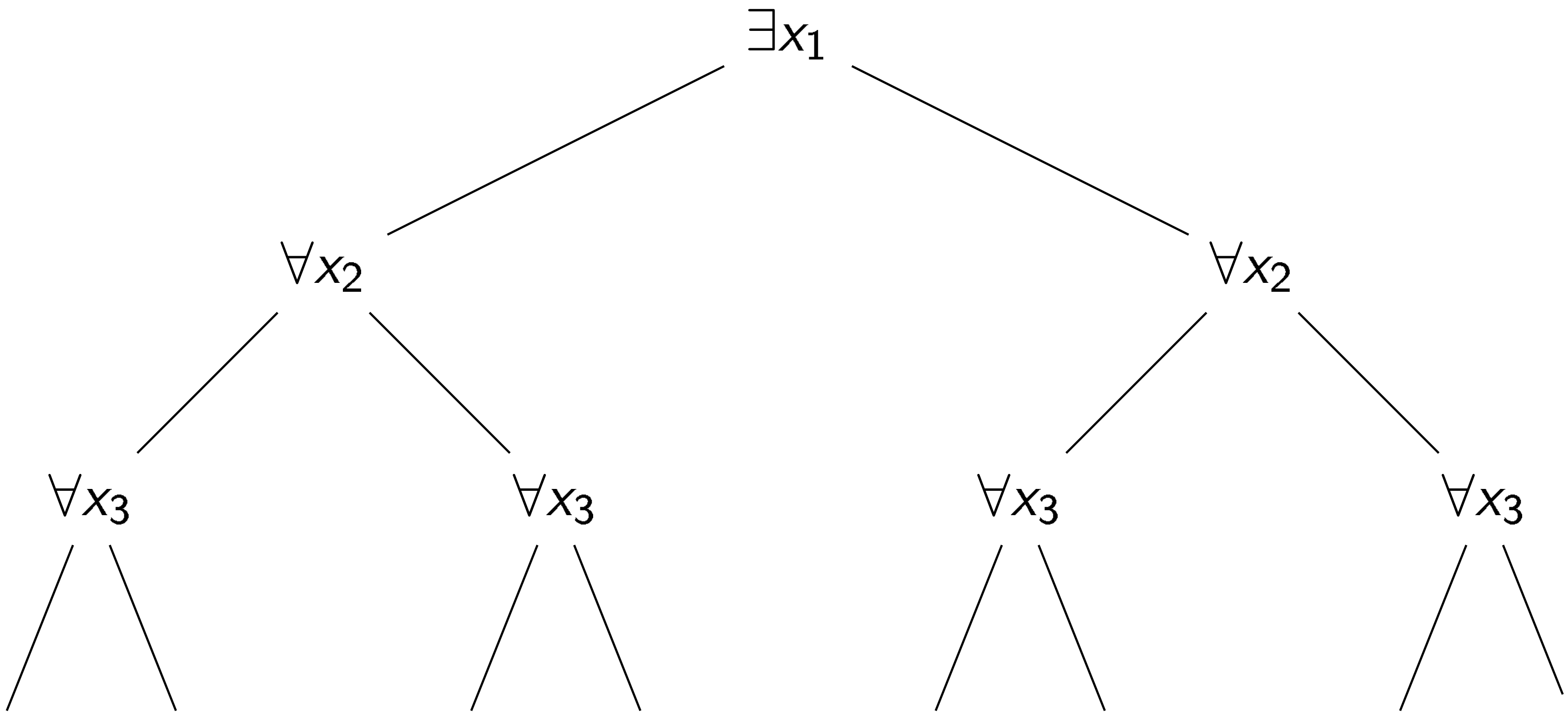
- $\mathtt{V}(\exists x\ \psi) = \mathtt{V}(\psi[x \leftarrow \text{false}]) \vee \mathtt{V}(\psi[x \leftarrow \text{true}])$
- $\mathtt{V}(\forall x\ \psi) = \mathtt{V}(\psi[x \leftarrow \text{false}]) \wedge \mathtt{V}(\psi[x \leftarrow \text{true}])$
- $\mathtt{V}(\beta) = \mathit{eval}(\beta)$ if β is a ground boolean formula.

Polynomial space but exponential time

Example

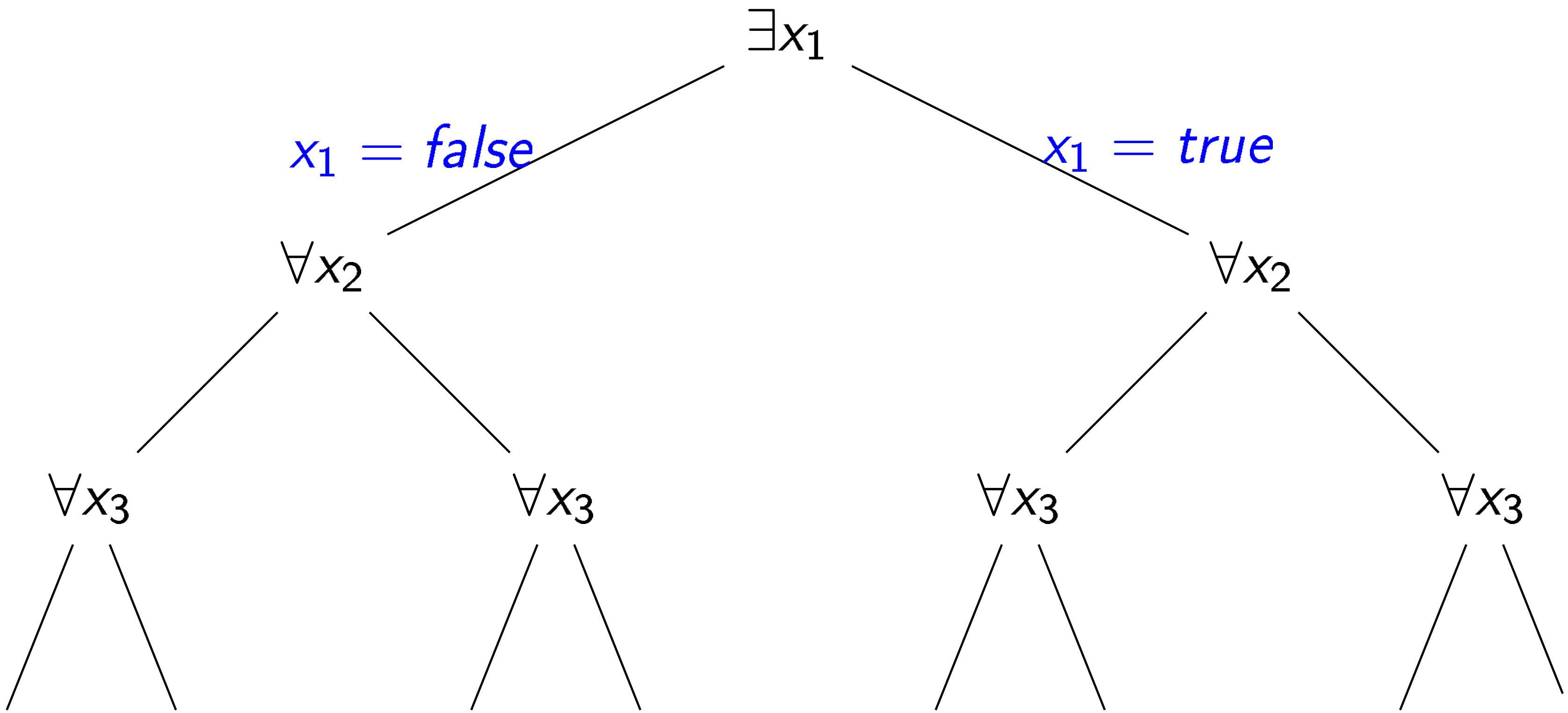
$$
\begin{aligned}
\mathtt{V}(\exists x_1 \forall x_2 \forall x_3\ \ x_1 \wedge (\neg x_2 \vee x_3)) &= \vee \begin{cases} \mathtt{V}(\forall x_2 \forall x_3\ \ \text{false} \wedge (\neg x_2 \vee x_3)) \\ \mathtt{V}(\forall x_2 \forall x_3\ \ \text{true} \wedge (\neg x_2 \vee x_3)) \end{cases} \\
&= \vee \begin{cases} \wedge \begin{cases} \mathtt{V}(\forall x_3\ \ \text{false} \wedge (\neg \text{false} \vee x_3)) \\ \mathtt{V}(\forall x_3\ \ \text{false} \wedge (\neg \text{true} \vee x_3)) \end{cases} \\ \wedge \begin{cases} \mathtt{V}(\forall x_3\ \ \text{true} \wedge (\neg \text{false} \vee x_3)) \\ \mathtt{V}(\forall x_3\ \ \text{true} \wedge (\neg \text{true} \vee x_3)) \end{cases} \end{cases} \\
&= \ldots
\end{aligned}
$$

Parallelistation scheme



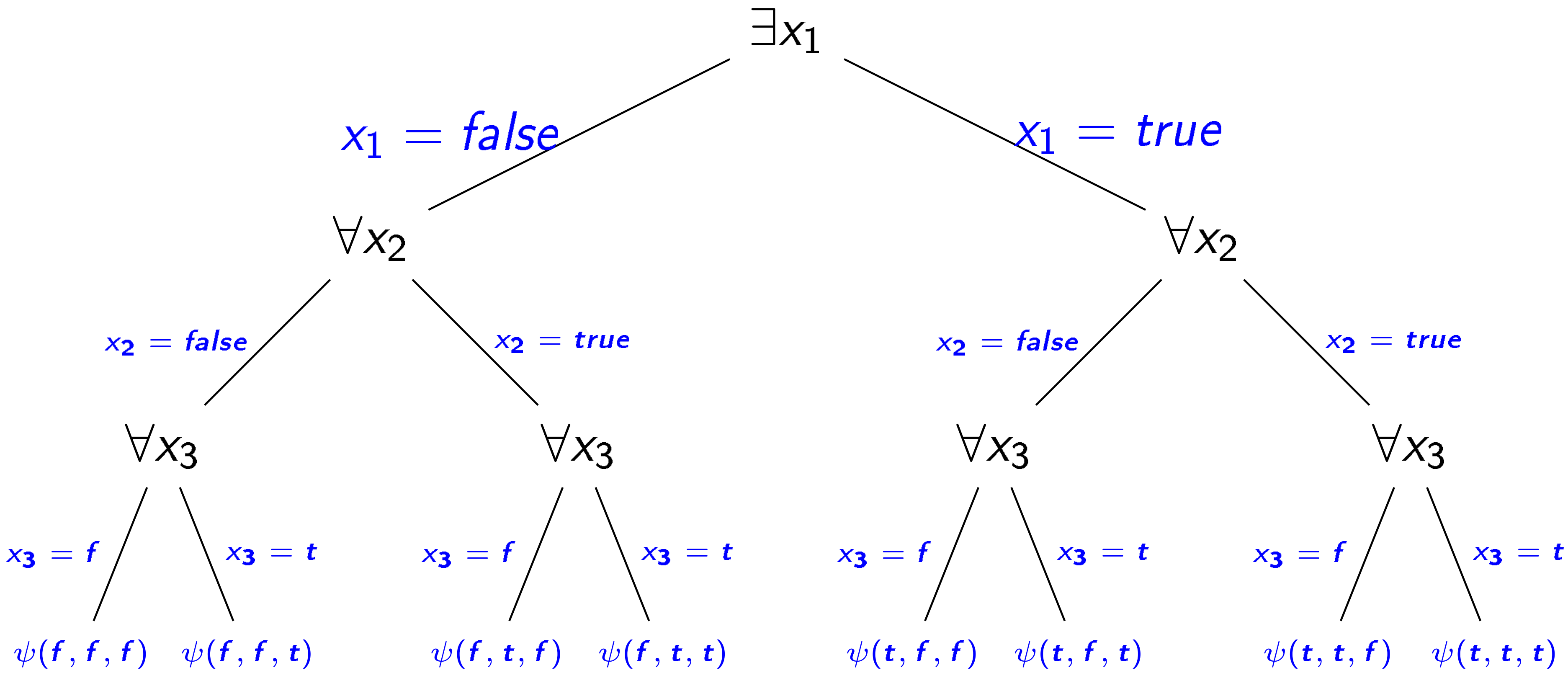
$\phi = \exists x_1 \forall x_2 \forall x_3 \;\; \psi(x_1, x_2, x_3)$ where $\psi = x_1 \wedge (\neg x_2 \vee x_3)$

Parallelistation scheme



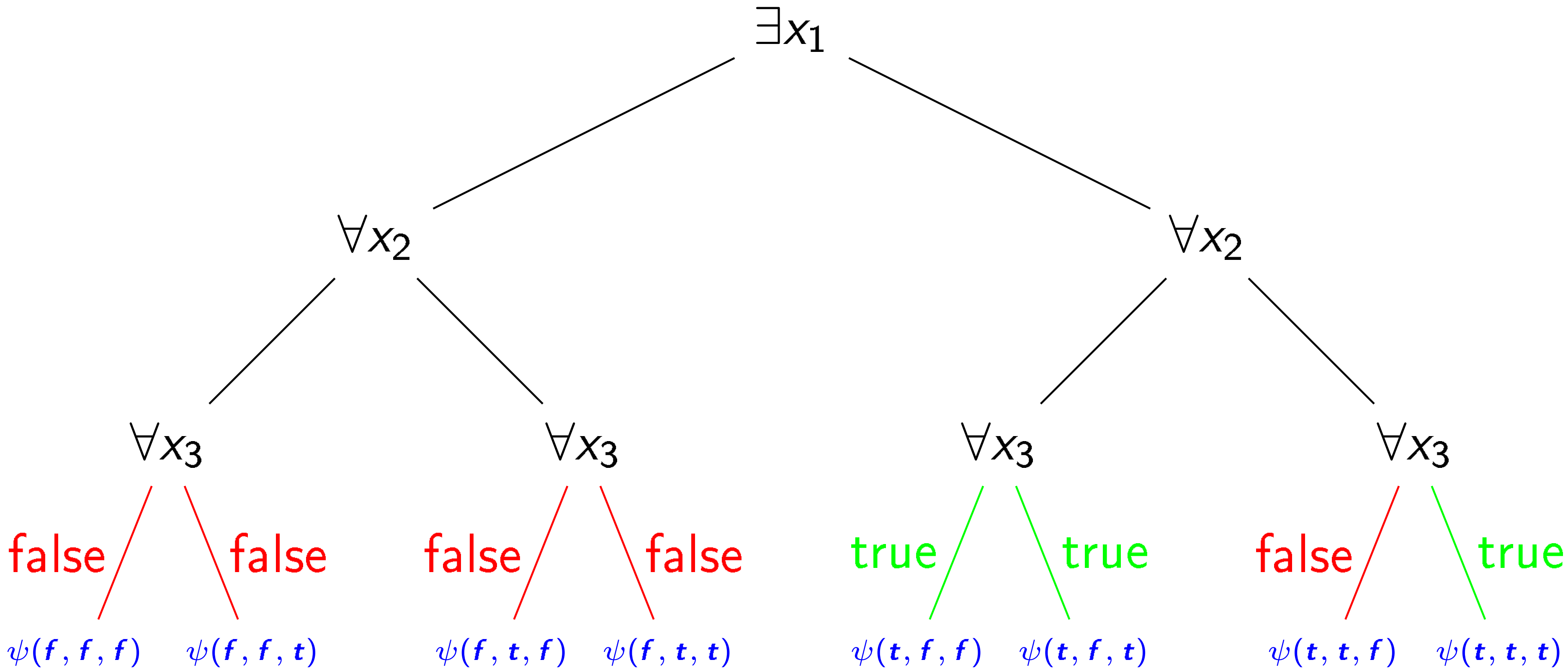
$$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3) \text{ where } \psi = x_1 \wedge (\neg x_2 \vee x_3)$$

Parallelistation scheme



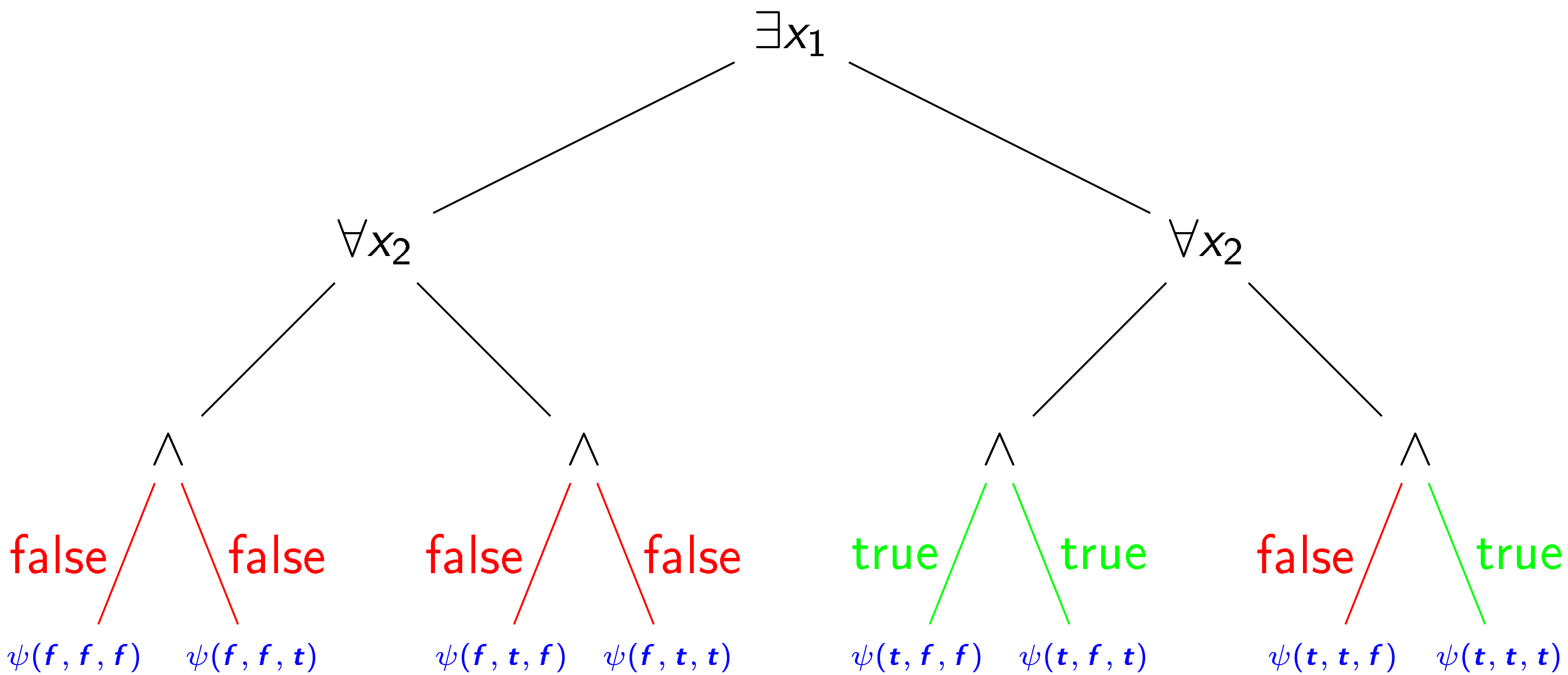
$$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3) \text{ where } \psi = x_1 \wedge (\neg x_2 \vee x_3)$$

Parallelistation scheme



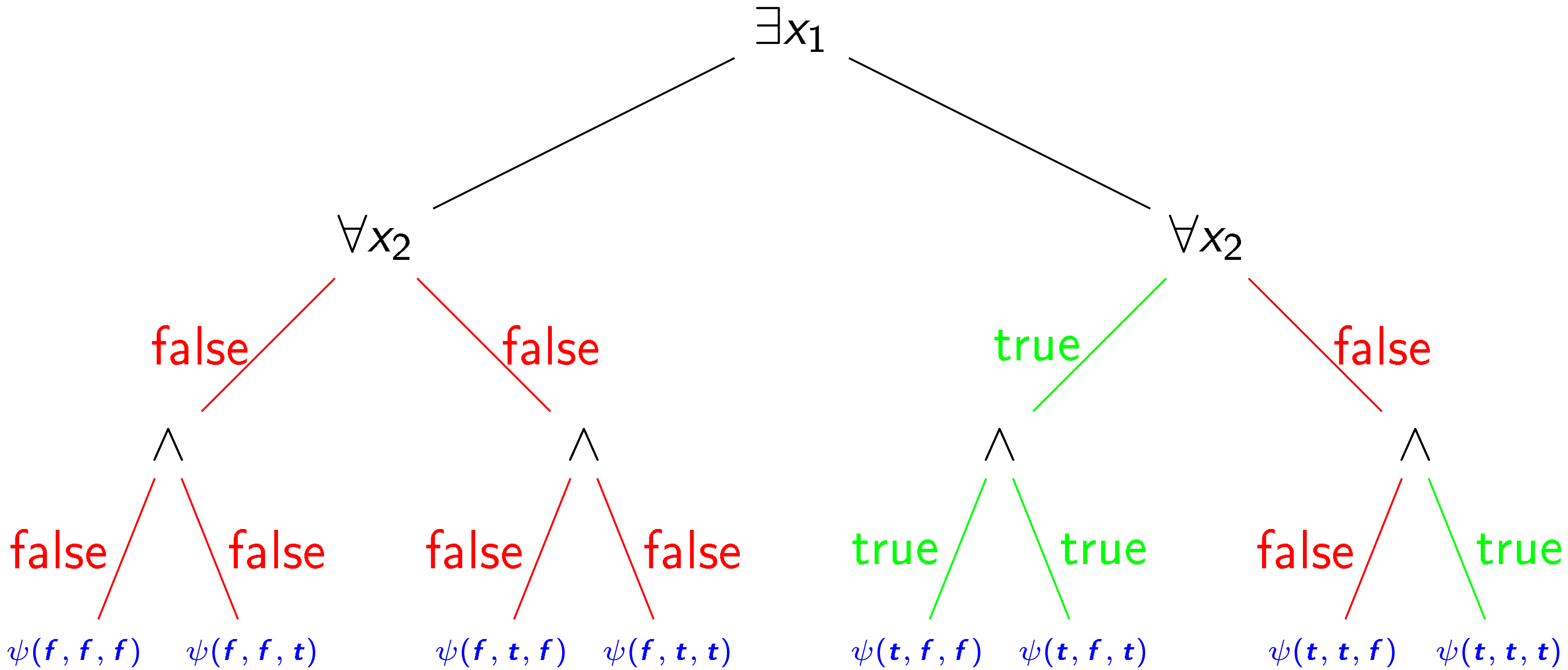
$$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3) \text{ where } \psi = x_1 \wedge (\neg x_2 \vee x_3)$$

Parallelistation scheme



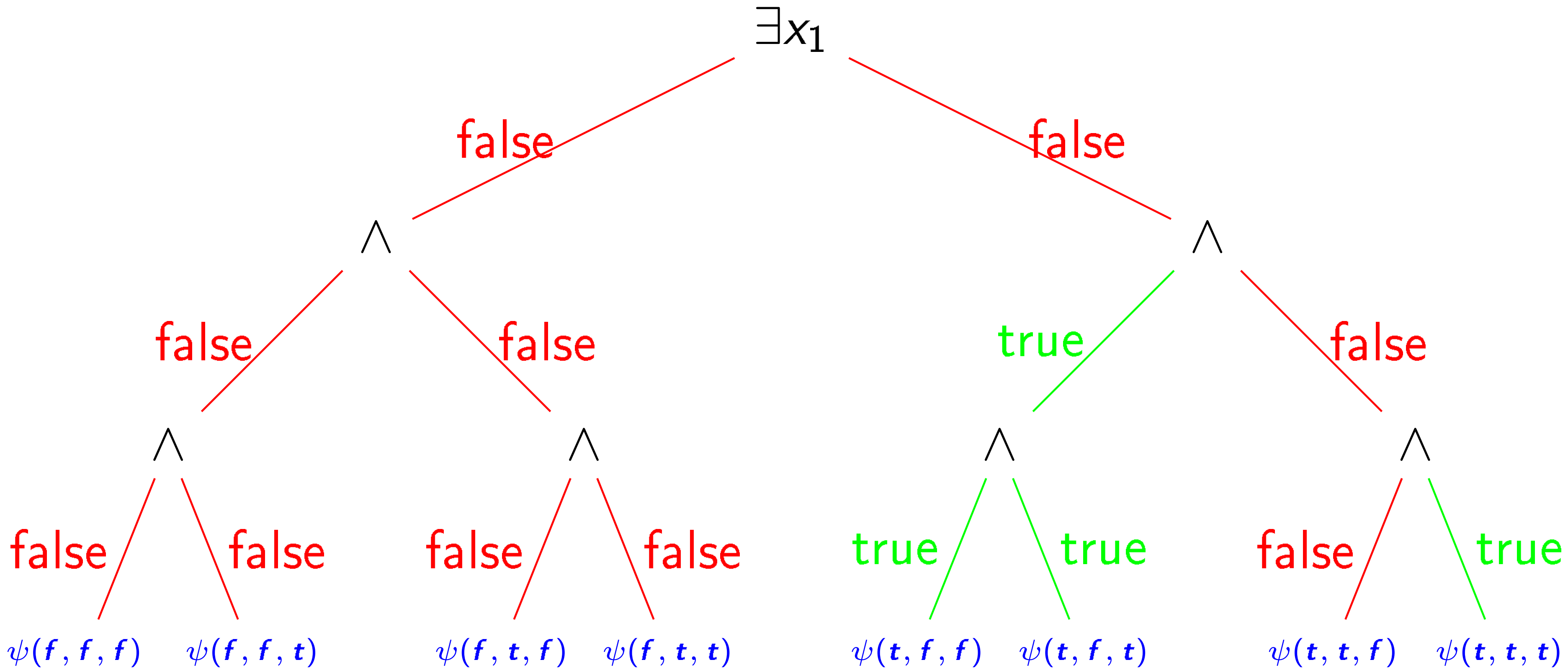
$\phi = \exists x_1 \forall x_2 \forall x_3 \;\; \psi(x_1, x_2, x_3)$ where $\psi = x_1 \wedge (\neg x_2 \vee x_3)$

Parallelistation scheme



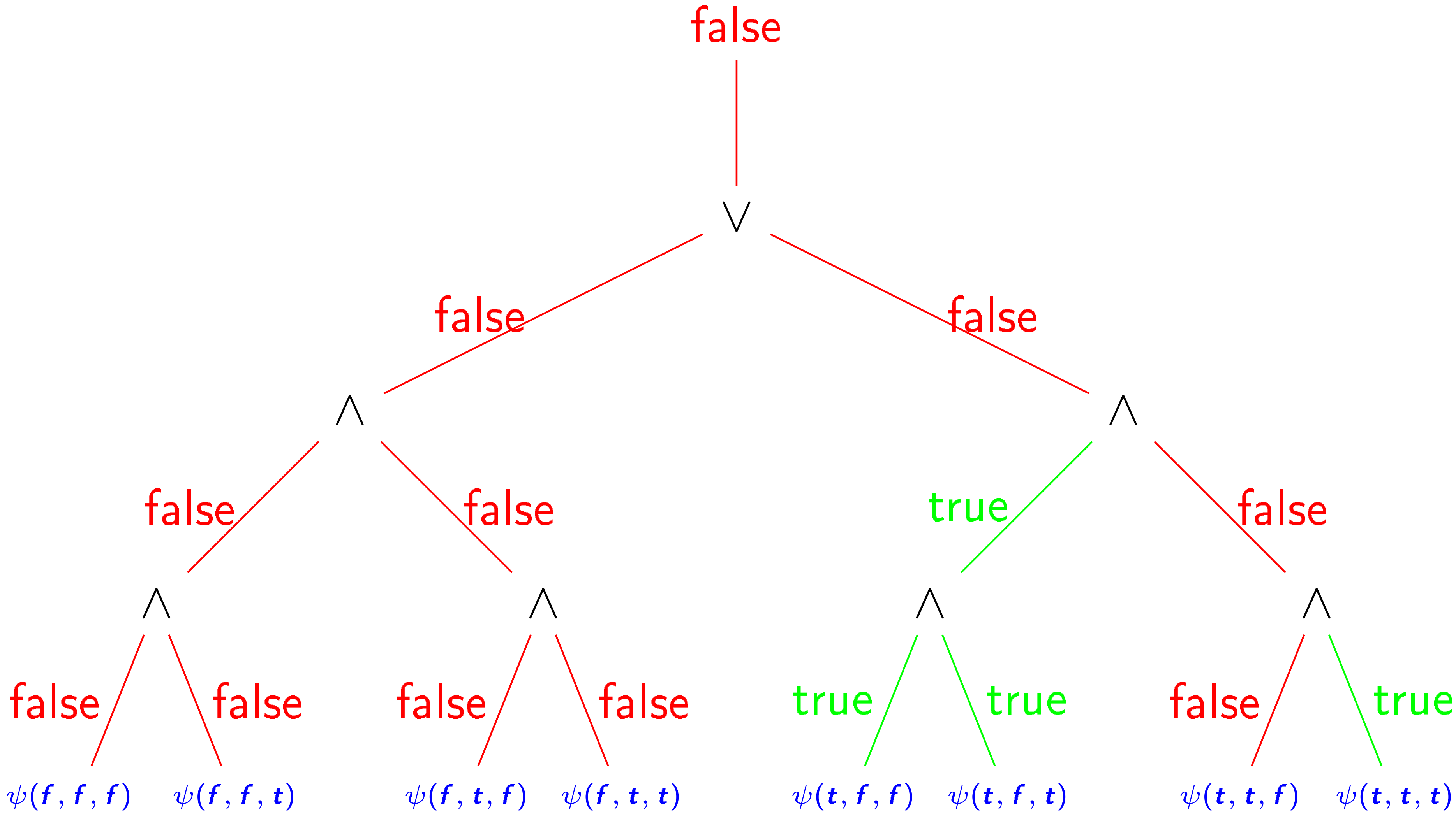
$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3)$ where $\psi = x_1 \wedge (\neg x_2 \vee x_3)$

Parallelistation scheme



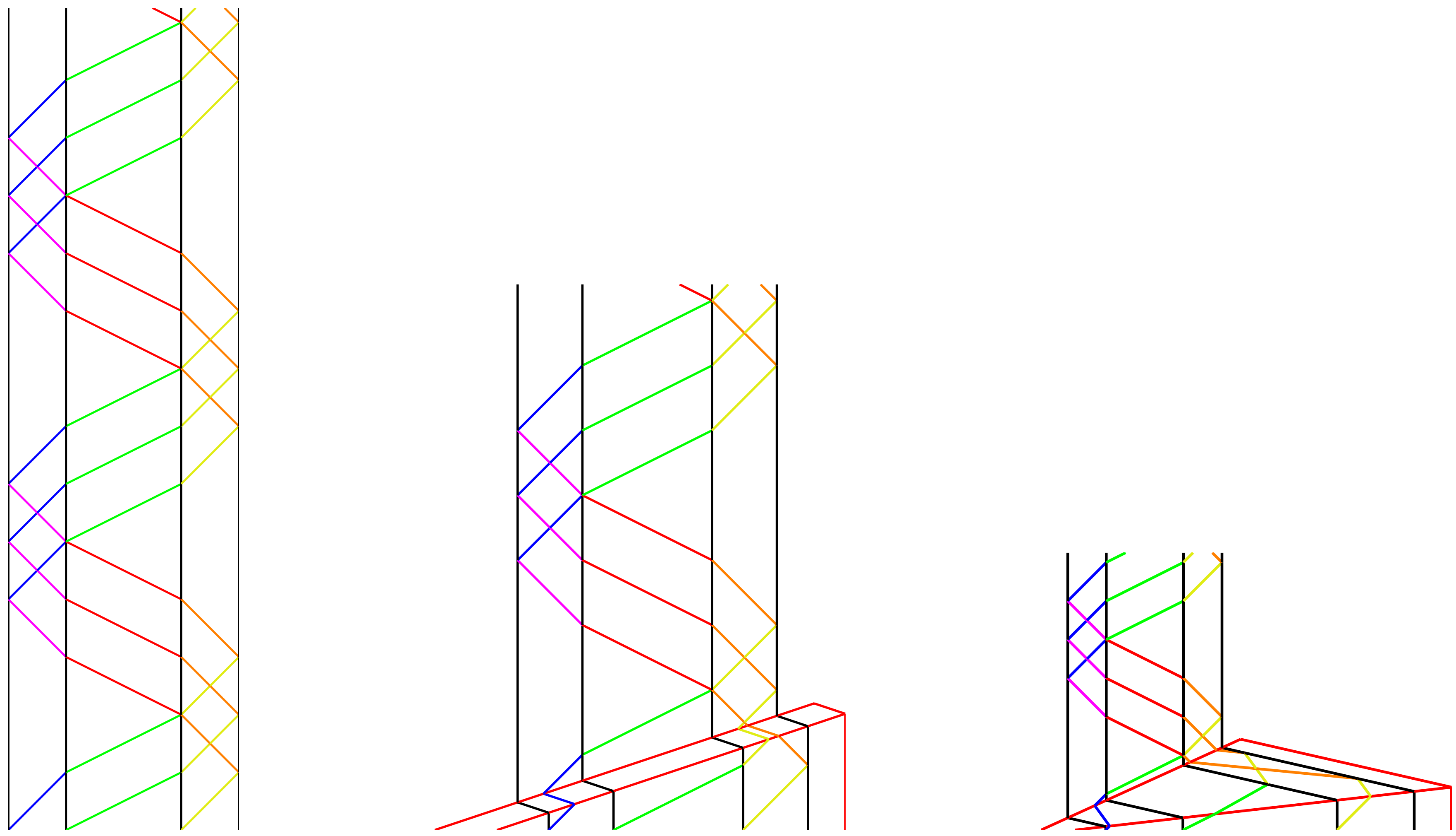
$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3)$ where $\psi = x_1 \wedge (\neg x_2 \vee x_3)$

Parallelistation scheme

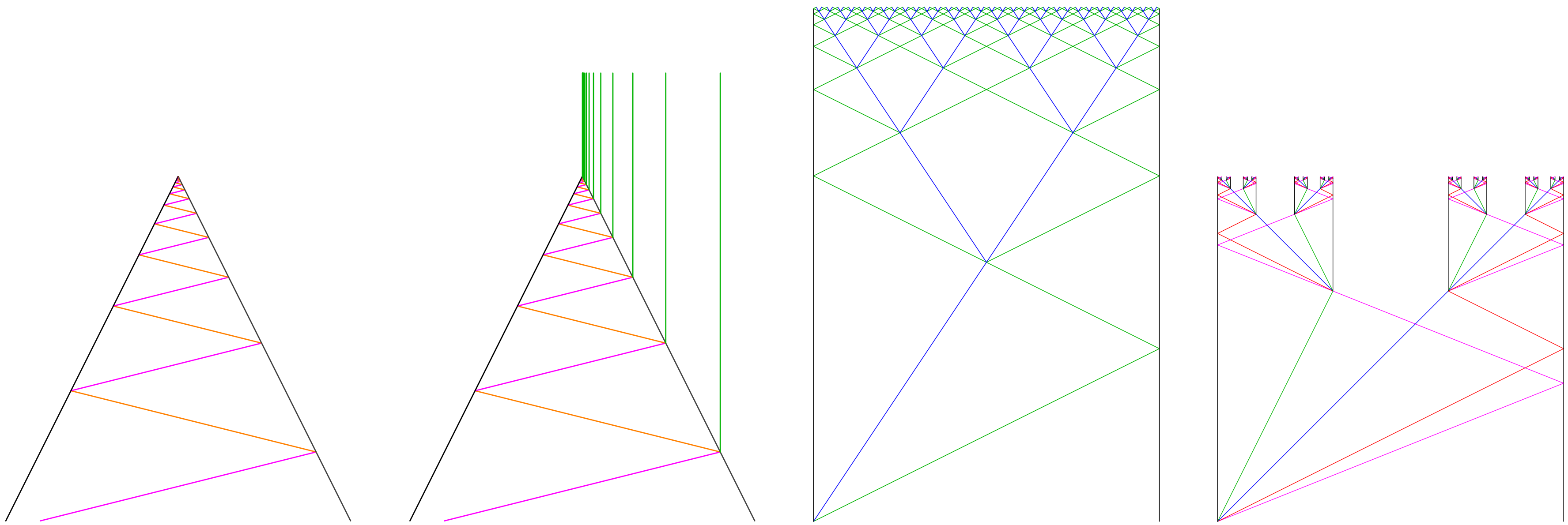


$$\phi = \exists x_1 \forall x_2 \forall x_3 \quad \psi(x_1, x_2, x_3) \text{ where } \psi = x_1 \wedge (\neg x_2 \vee x_3)$$

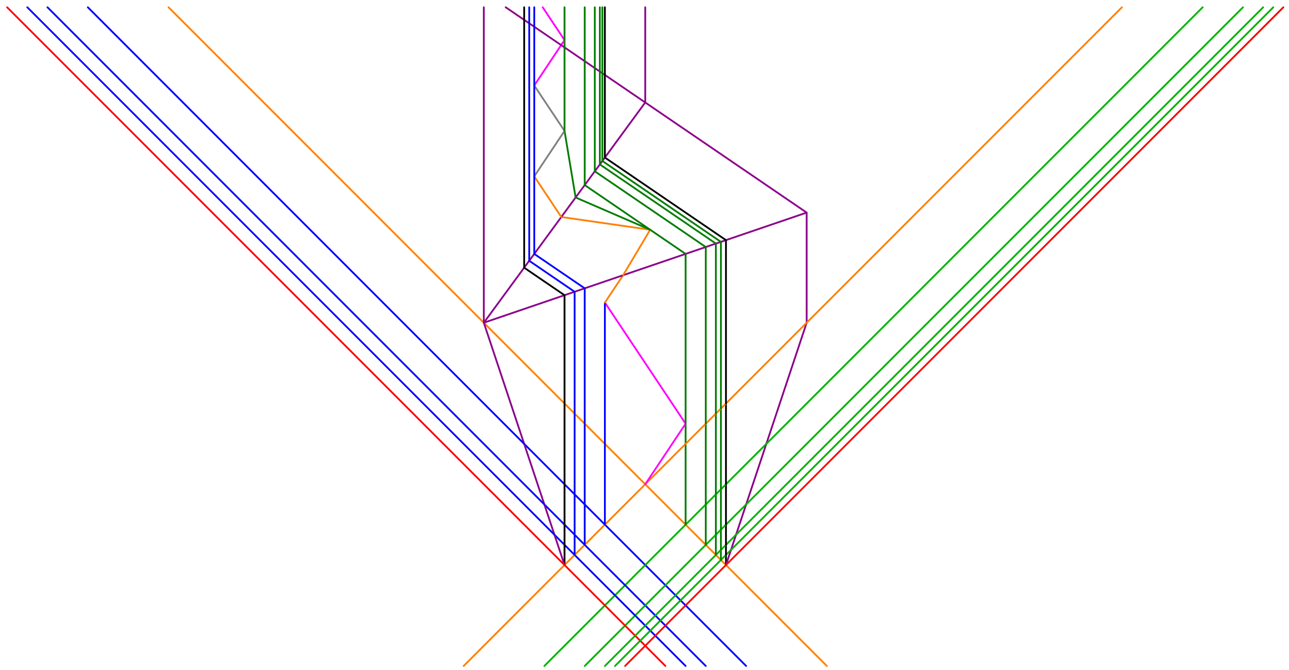
"Nice regular drawings"



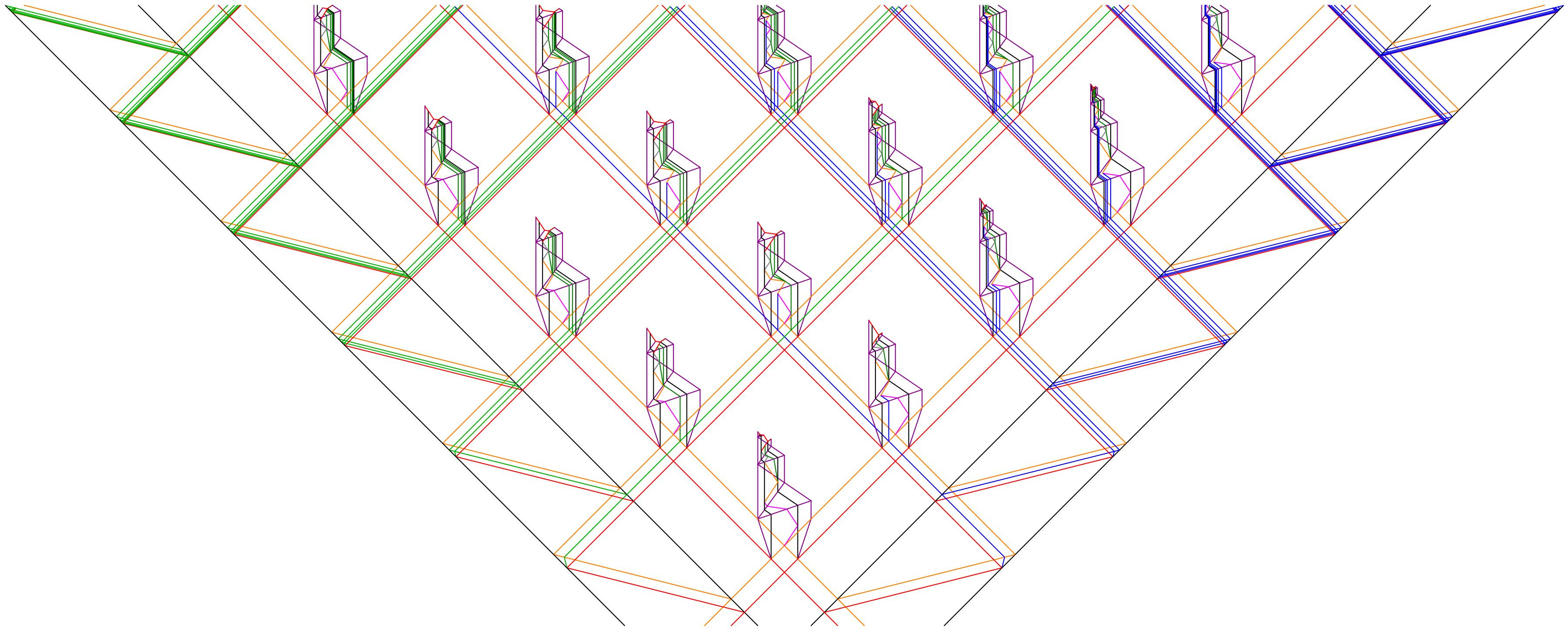
"Nice regular drawings"



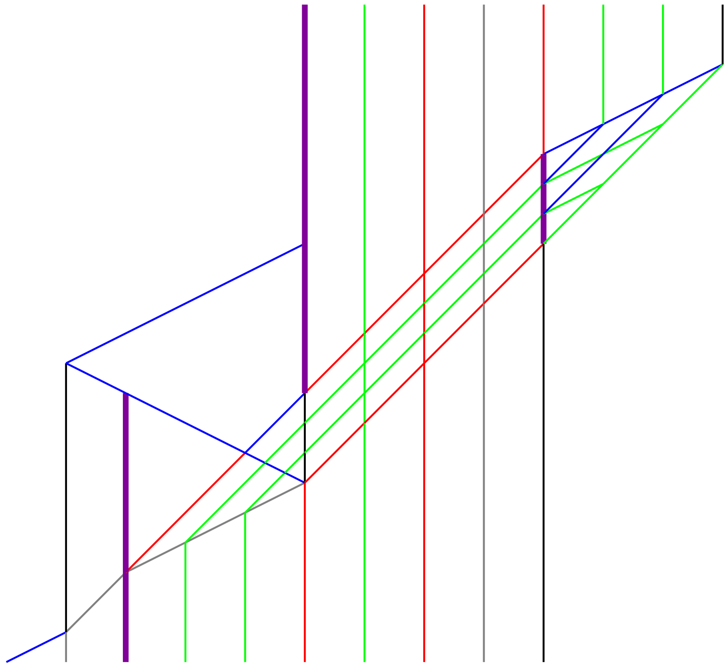
"Nice regular drawings"

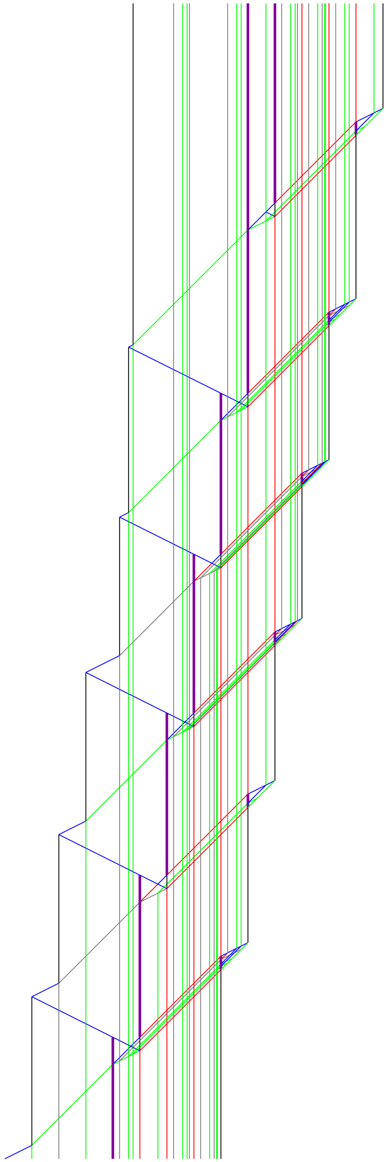


"Nice regular drawings"



“Nice regular drawings”

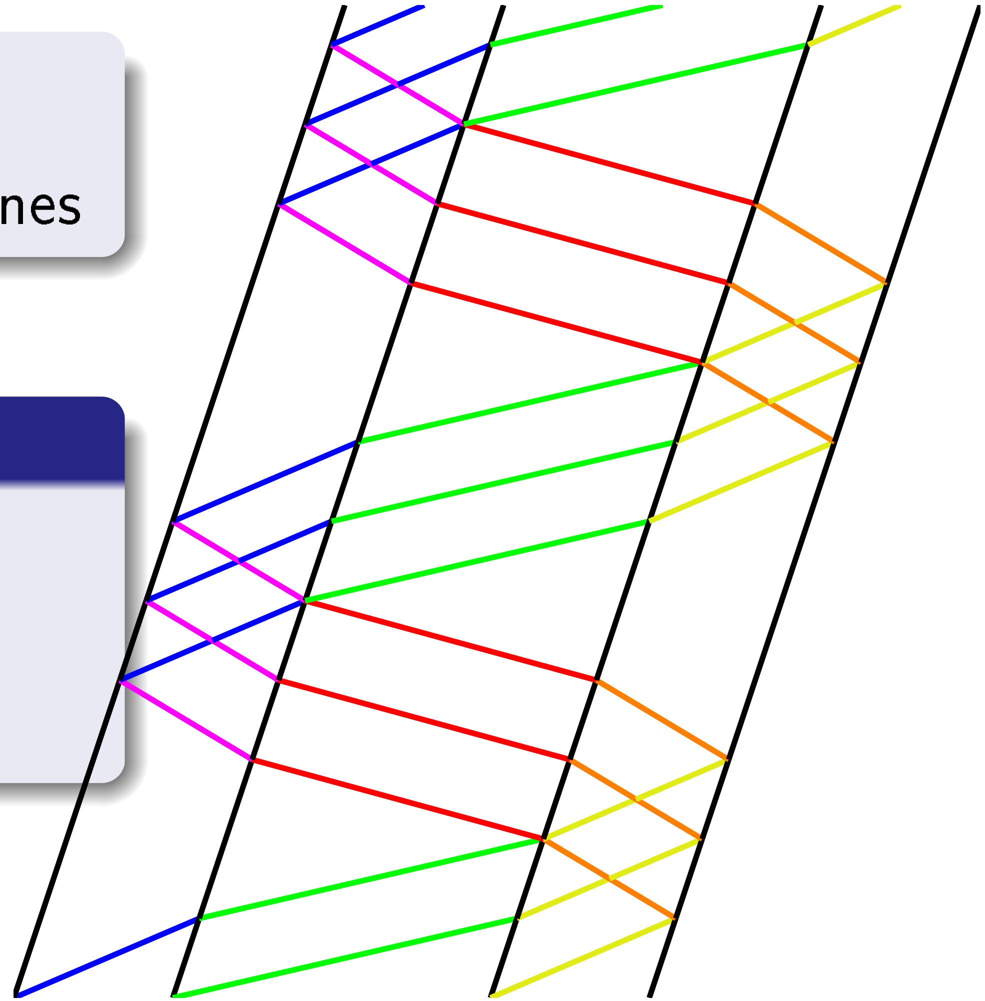




"Nice regular drawings"

Lines: traces of signals

Space-time diagrams of signal machines



Defined by

- bottom: initial configuration
- lines: signals $\rightsquigarrow$ meta-signals
- end-points: collisions $\rightsquigarrow$ rules

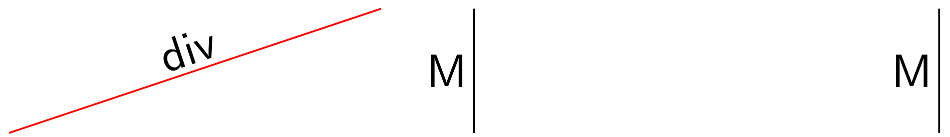
Example: find the middle

Meta-signals (speed)

M (0)

Collision rules

Example: find the middle

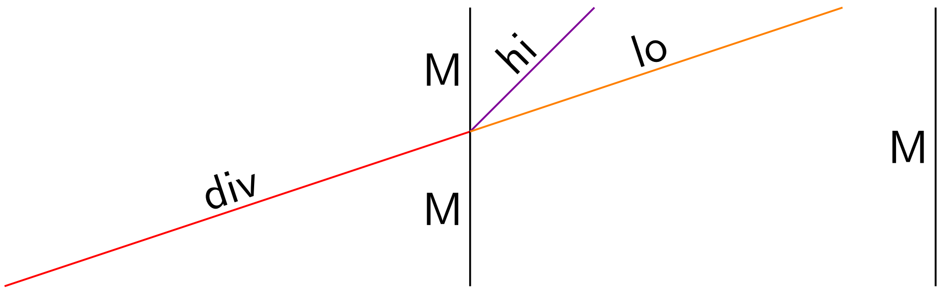


Meta-signals (speed)

M (0)
div (3)

Collision rules

Example: find the middle



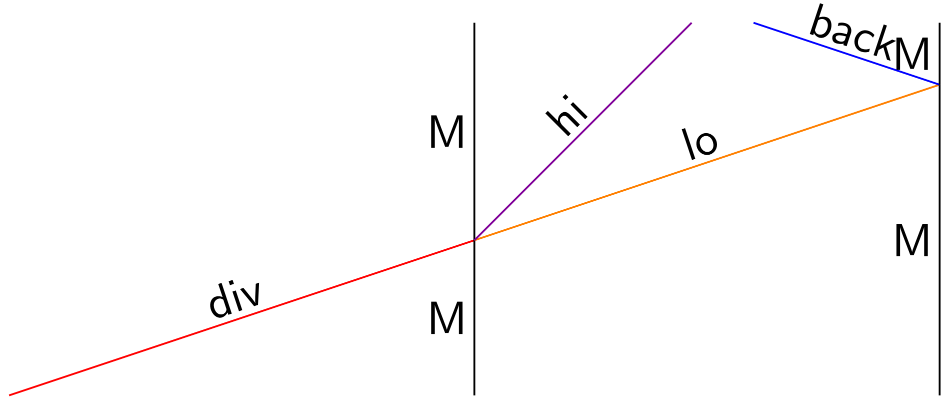
Meta-signals (speed)

- M (0)
- div (3)
- hi (1)
- lo (3)

Collision rules

{ div, M } $\rightarrow$ { M, hi, lo }

Example: find the middle



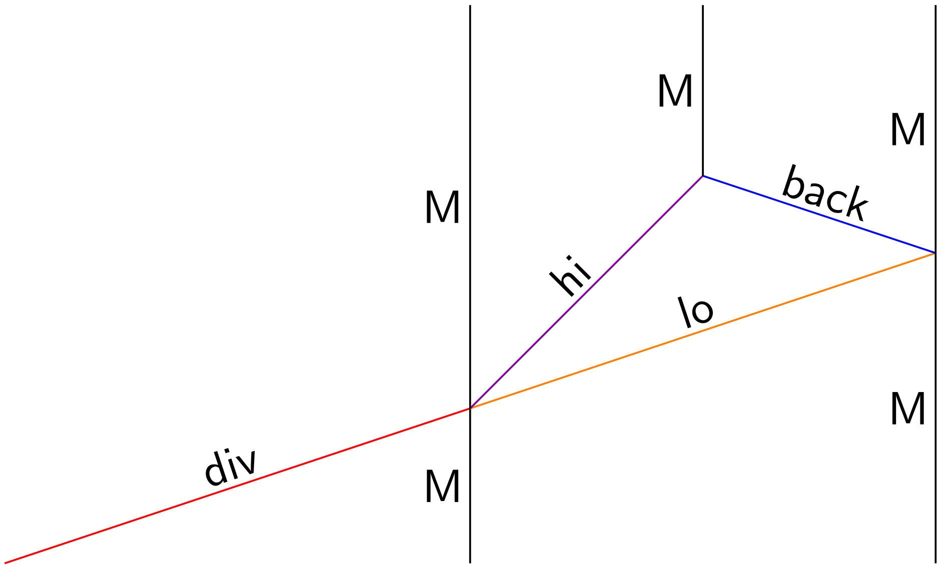
Meta-signals (speed)

M (0)
div (3)
hi (1)
lo (3)
back (-3)

Collision rules

{ div, M } → { M, hi, lo }
{ lo, M } → { back, M }

Example: find the middle



Meta-signals (speed)

- M (0)
- div (3)
- hi (1)
- lo (3)
- back (-3)

Collision rules

{ div, M } → { M, hi, lo }
{ lo, M } → { back, M }
{ hi, back } → { M }

Known results

Turing computation

- [Durand-Lose, 2011]

Known results

Turing computation

- [Durand-Lose, 2011]

Analog computations

- Computable analysis [Weihrauch, 2000] [Durand-Lose, 2010]
- Blum, Shub and Smale model [Blum et al., 1989] [Durand-Lose, 2008]

Known results

Turing computation

- [Durand-Lose, 2011]

Analog computations

- Computable analysis [Weihrauch, 2000] [Durand-Lose, 2010]
- Blum, Shub and Smale model [Blum et al., 1989] [Durand-Lose, 2008]

"Black hole" implementation

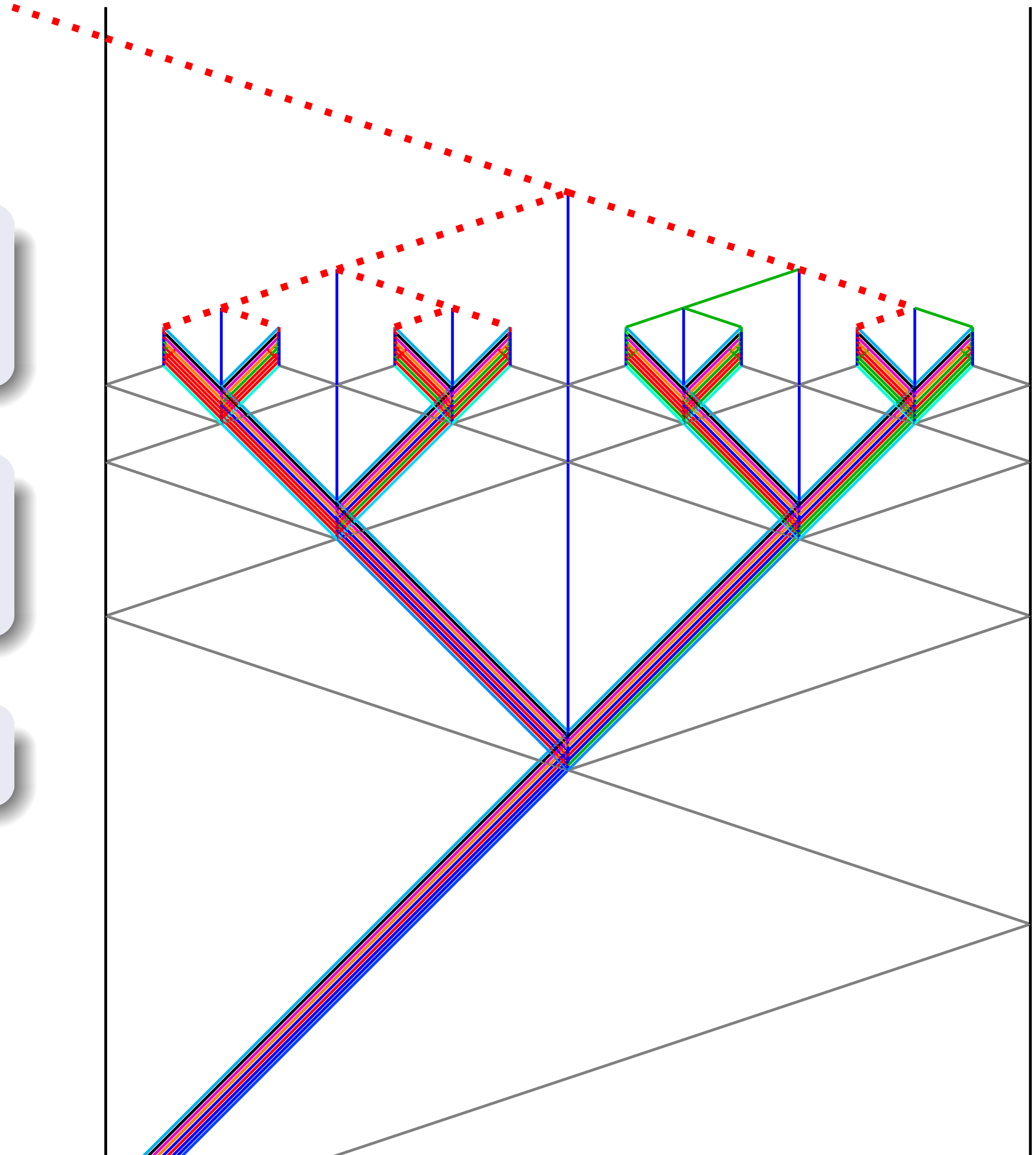
- Hyper-computation, deciding the Halting problem [Durand-Lose, 2009]

Whole picture

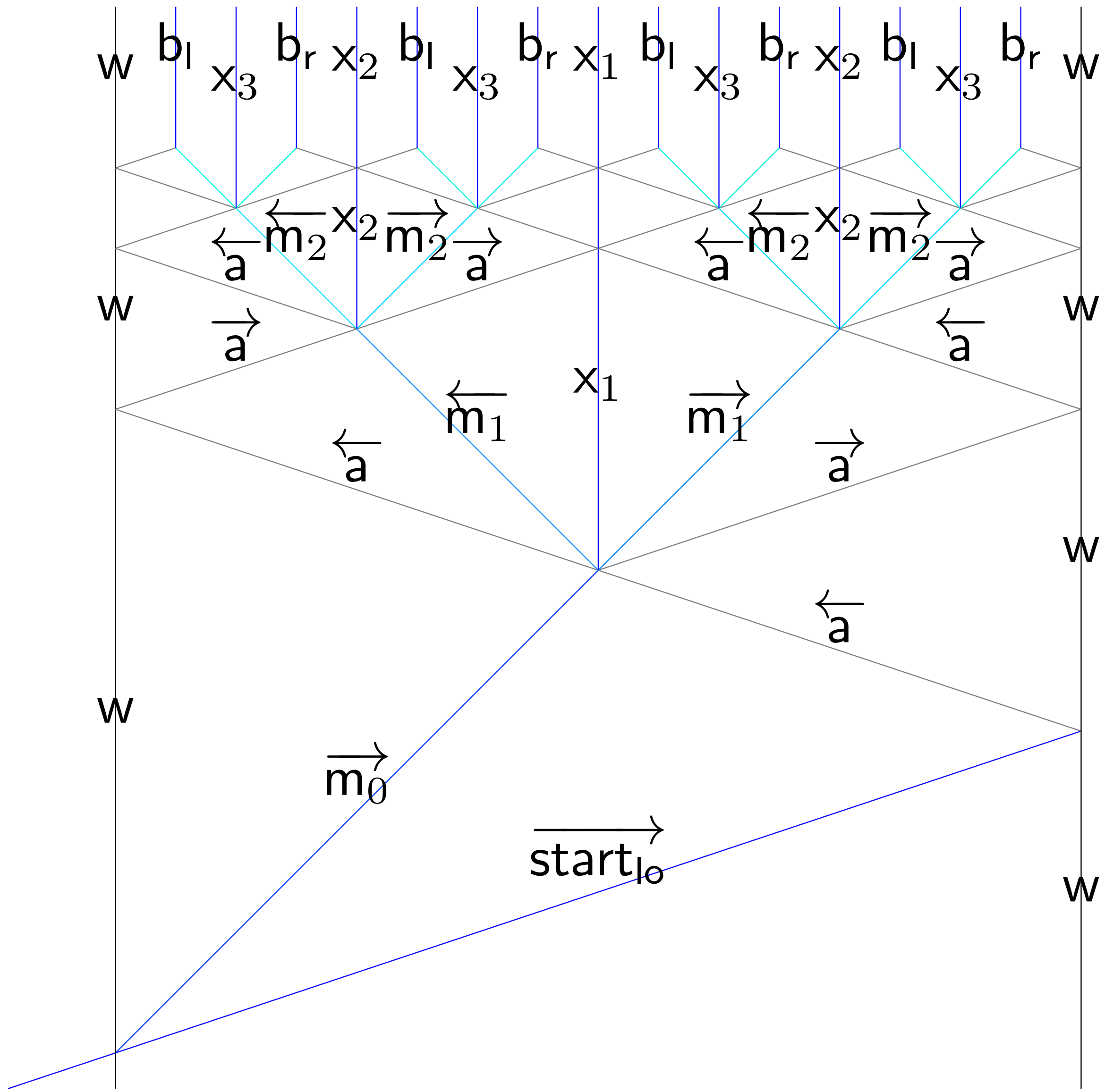
Q-formula compiled into a signal machine

Linear number of meta-signals

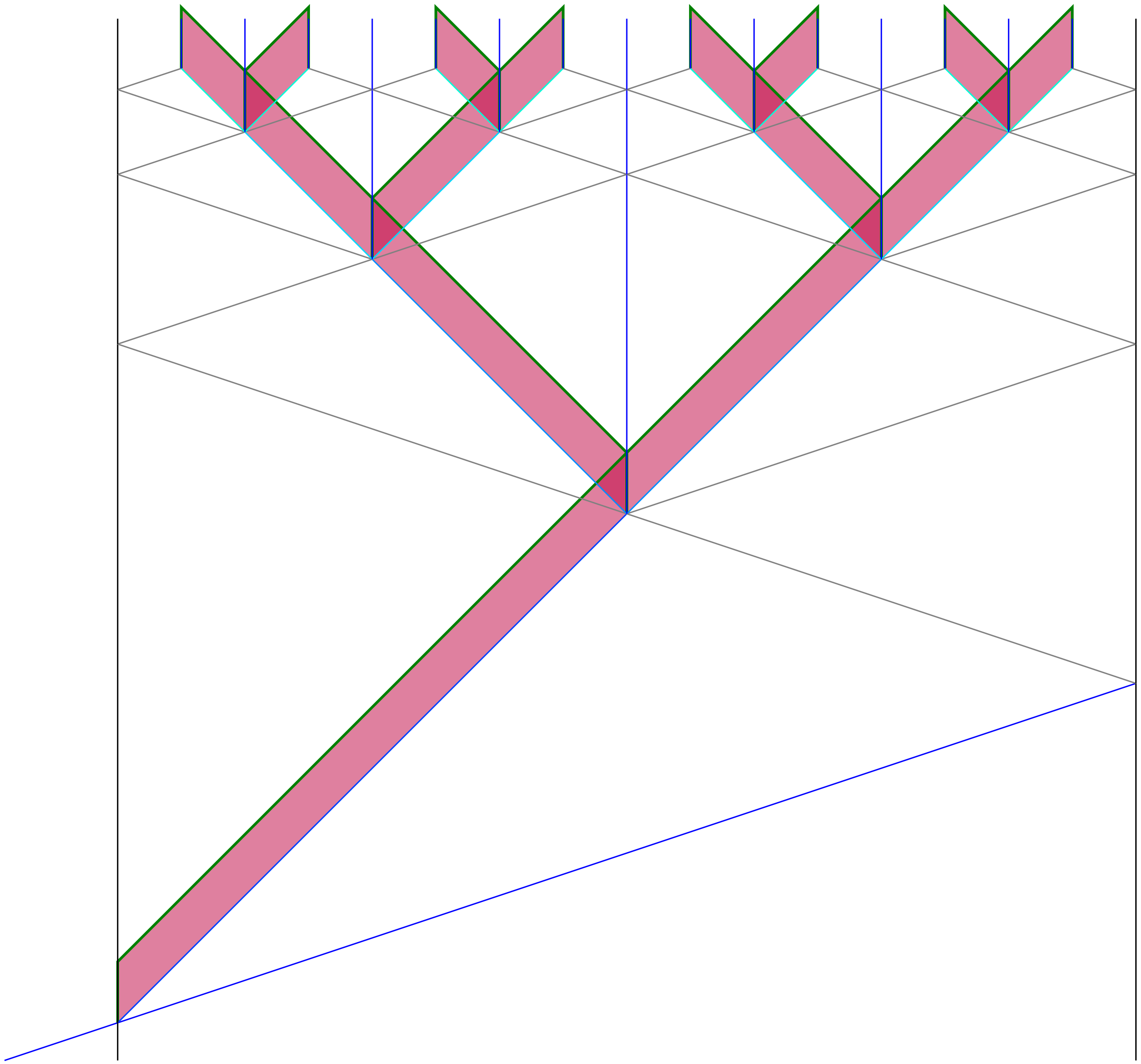
Quadratic number of rules



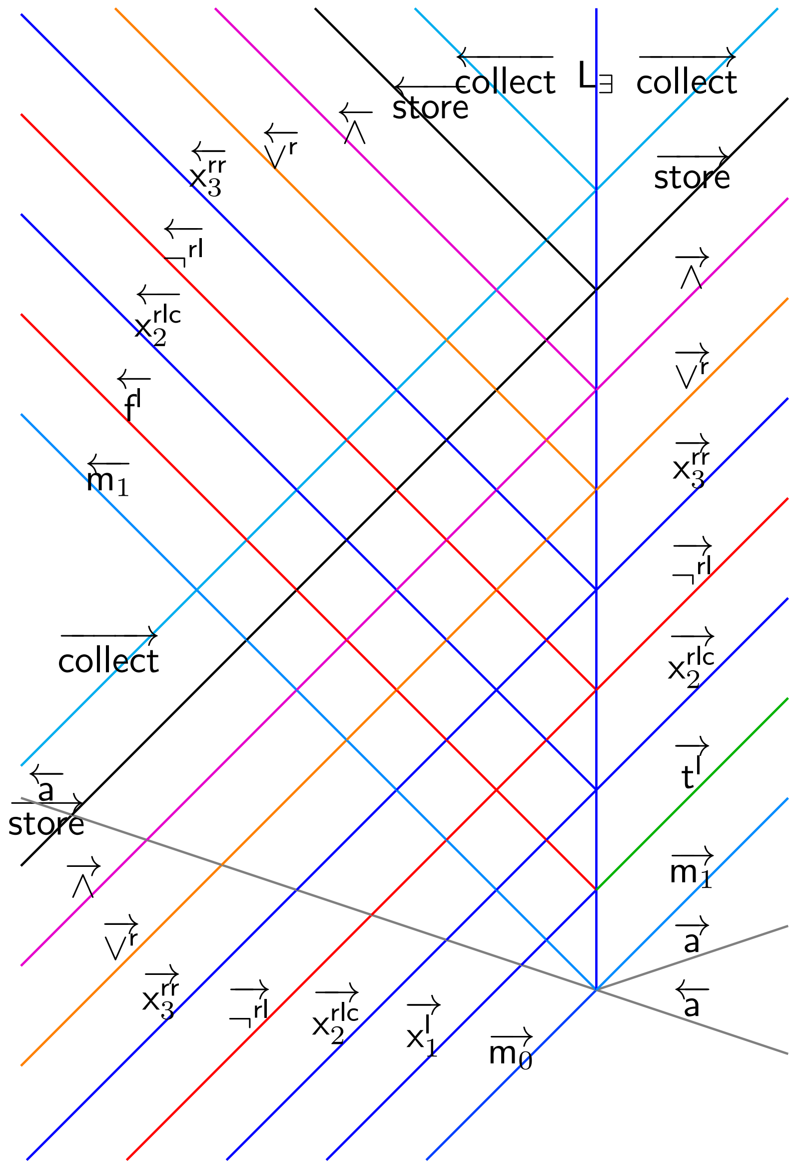
Building the tree with signals



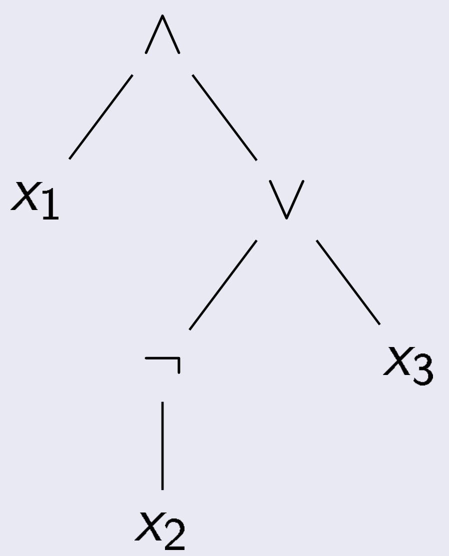
Following the tree



Propagating the beam

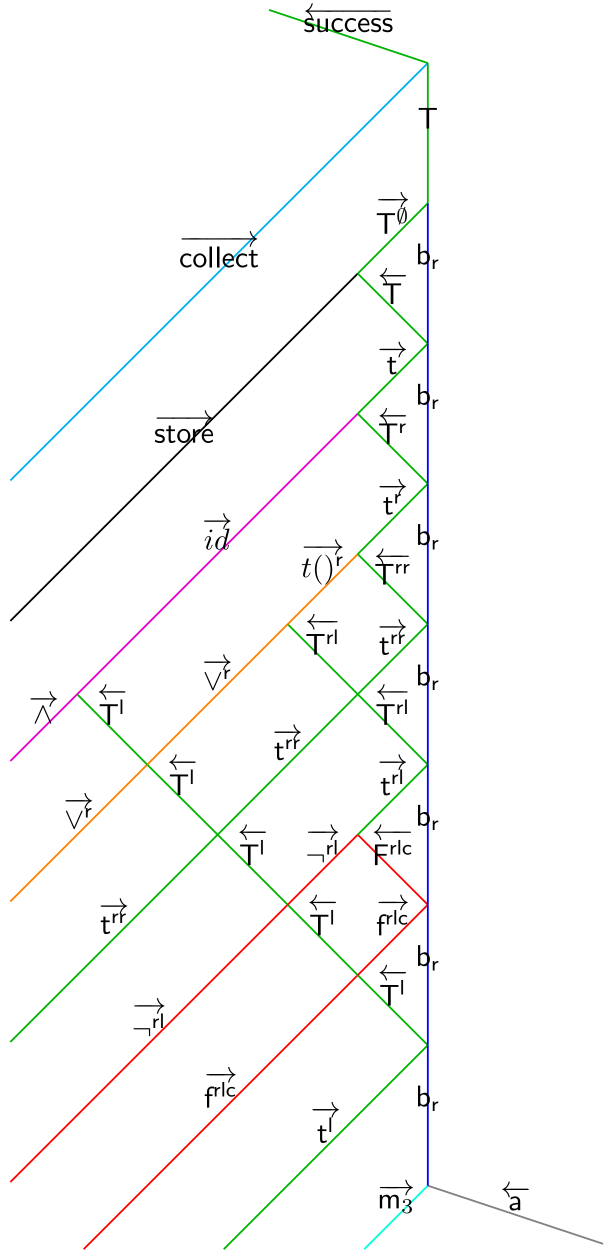


Evaluating the formula

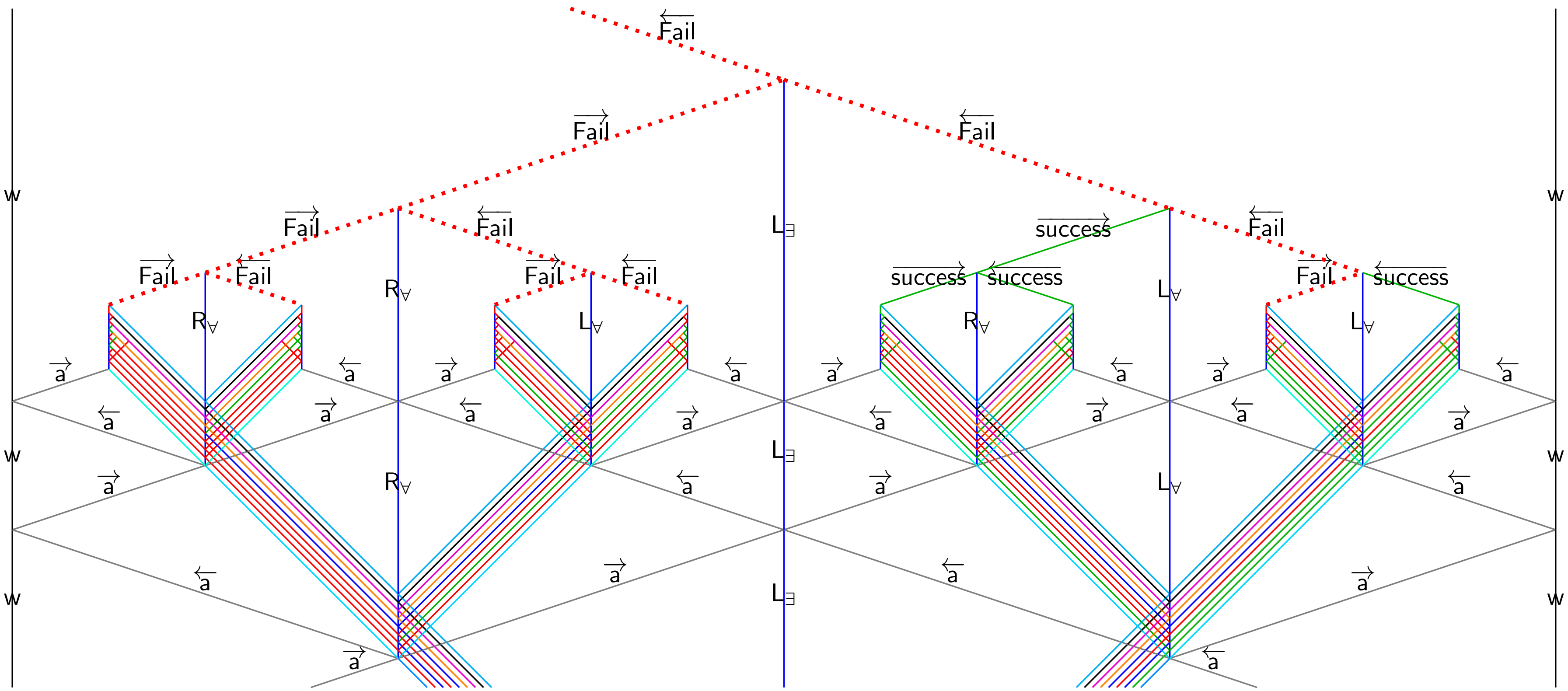


case:

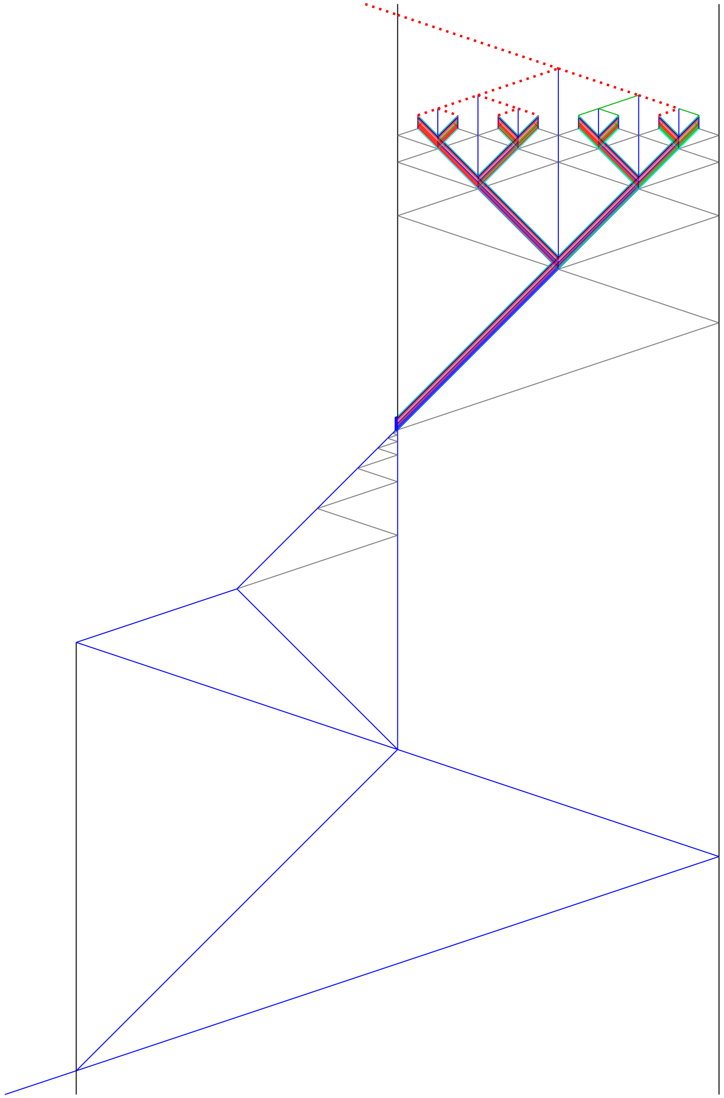
- x_1 is true
- x_2 is false
- x_3 is true



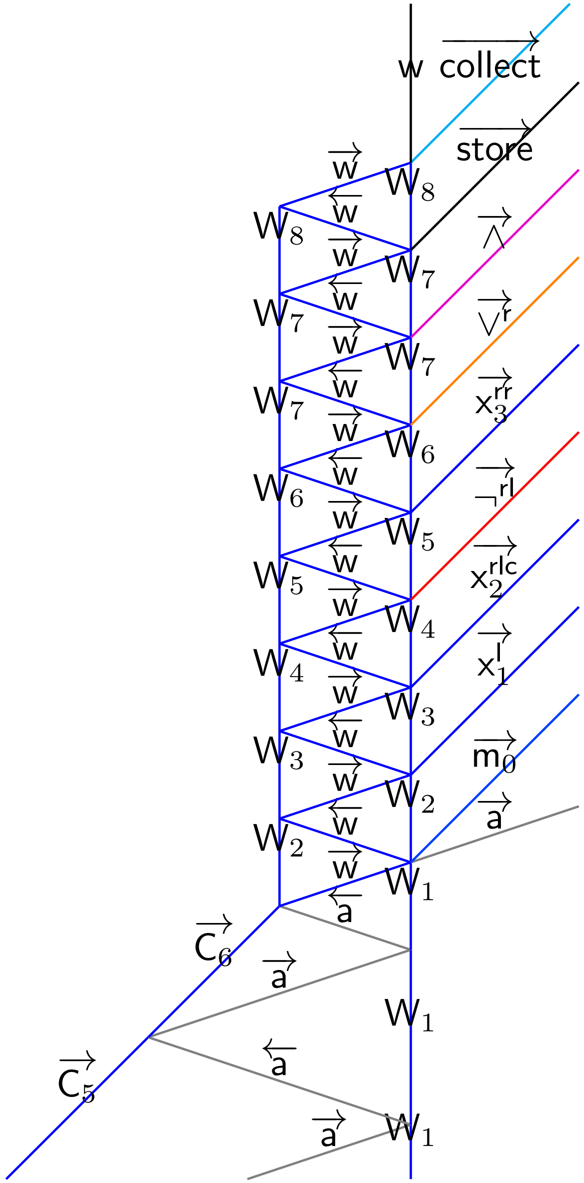
Collecting the results



Whole picture with initialization



Generating the formula



Results

- QSAT resolution in bounded space and time
- Quadratic depth
- Exponential width

Perspectives

- Generic version
- Model of the parallel thesis
- Higher complexity classes

Blum, L., Shub, M., and Smale, S. (1989).
On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines.
Bull. Amer. Math. Soc., 21(1):1–46.

Durand-Lose, J. (2008).
Abstract geometrical computation with accumulations: Beyond the Blum, Shub and Smale model.
In Beckmann, A., Dimitracopoulos, C., and Löwe, B., editors, *Logic and Theory of Algorithms, 4th Conf. Computability in Europe (CiE '08) (abstracts and extended abstracts of unpublished papers)*, pages 107–116. University of Athens.

Durand-Lose, J. (2009).
Abstract geometrical computation 3: Black holes for classical and analog computing.
Nat. Comput., 8(3):455–472.

Durand-Lose, J. (2010).

Abstract geometrical computation 5: embedding computable analysis.

Nat. Comput.

Special issue on Unconv. Comp. '09.

Durand-Lose, J. (2011).

Abstract geometrical computation 4: small Turing universal signal machines.

Theoret. Comp. Sci., 412:57–67.

Weihrauch, K. (2000).

Introduction to computable analysis.

Texts in Theoretical computer science. Springer, Berlin.