Online Parametric Timed Pattern Matching with Automata-Based Skipping

Masaki Waga[1,2,3] and Étienne André[4,5,1]

National Institute of Informatics[1], SOKENDAI[2],
JSPS Research Fellow[3],
LIPN, Université Paris 13, CNRS[4], JFLI, UMI CNRS[5]

15 Apr. 2019, MT-CPS 2019
Accepted to NFM 2019

This work is partially supported by JST ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), by JSPS Grants-in-Aid No. 15KT0012 & 18J22498 and by the ANR national research program PACS (ANR-14-CE28-0002).

Parameterized Specification

Monitoring

Online Parametric Timed Pattern Matching with Automata-Based Skipping

Masaki Waga[1,2,3] and Étienne André[4,5,1]

National Institute of Informatics[1], SOKENDAI[2],
JSPS Research Fellow[3],
LIPN, Université Paris 13, CNRS[4], JFLI, UMI CNRS[5]

15 Apr. 2019, MT-CPS 2019
Accepted to NFM 2019

This work is partially supported by JST ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), by JSPS Grants-in-Aid No. 15KT0012 & 18J22498 and by the ANR national research program PACS (ANR-14-CE28-0002).

(Non-Parametric) Timed Pattern Matching

[Ulus+, FORMATS'14, Waga+, FORMATS'16]

Input

- **Time-series data**
  - System ***log***
    - e.g., change of engine rotation (ω) / velocity (v) of a car
      $v\uparrow$ at 0.1s, $\omega\downarrow$ at 0.2s, …
- **Real-time spec.**
  - ***Spec.*** useful for debugging
    - e.g., unexpected behavior of a car
      ω gets high and remains $\Rightarrow$ v gets high > 2 s. later

Output

- The intervals where the ***spec.*** is satisfied in the ***log***
  - e.g., The above behavior occurs in 0.8s-3.4s

Parametric Timed Pattern Matching

[André, Hasuo, & Waga, ICECCS'18]

Input

- **Time-series data**
  - System ***log***
    - e.g., change of engine rotation (ω) / velocity (v) of a car
      $v_\uparrow$ at 0.1s, $\omega_\downarrow$ at 0.2s, …
- ***Parametric*** **Real-time spec.**
  - ***Spec.*** useful for debugging (with **parameters**)
    - e.g., unexpected behavior of a car (with **parameters**)
      ω gets high and remains ⇒ v gets high p s. later

Output

- The intervals **+ param. valuation**, s.t. the ***spec.*** is satisfied in the ***log***
  - e.g., The above behavior occurs in 0.8s-3.4s, $p = 2.5$

Parametric Timed Pattern Matching

[André, Hasuo, & Waga, ICECCS'18]

Input

- **Time-series data**
  - System ***log***
    - e.g., change of engine rotation (ω) / velocity (v) of a car
      $v\uparrow$ at 0.1s, $\omega\downarrow$ at 0.2s, …
- ***Parametric*** **Real-time spec.**
  - ***Spec.*** useful for debugging (with **parameters**)
    - e.g., unexpected behavior of a car (with **parameters**)
      ω gets high and remains ⇒ v gets high p s. later

$p > 2$: satisfied (unexpected beh.)
$p \leq 2$: violated (expected beh.)

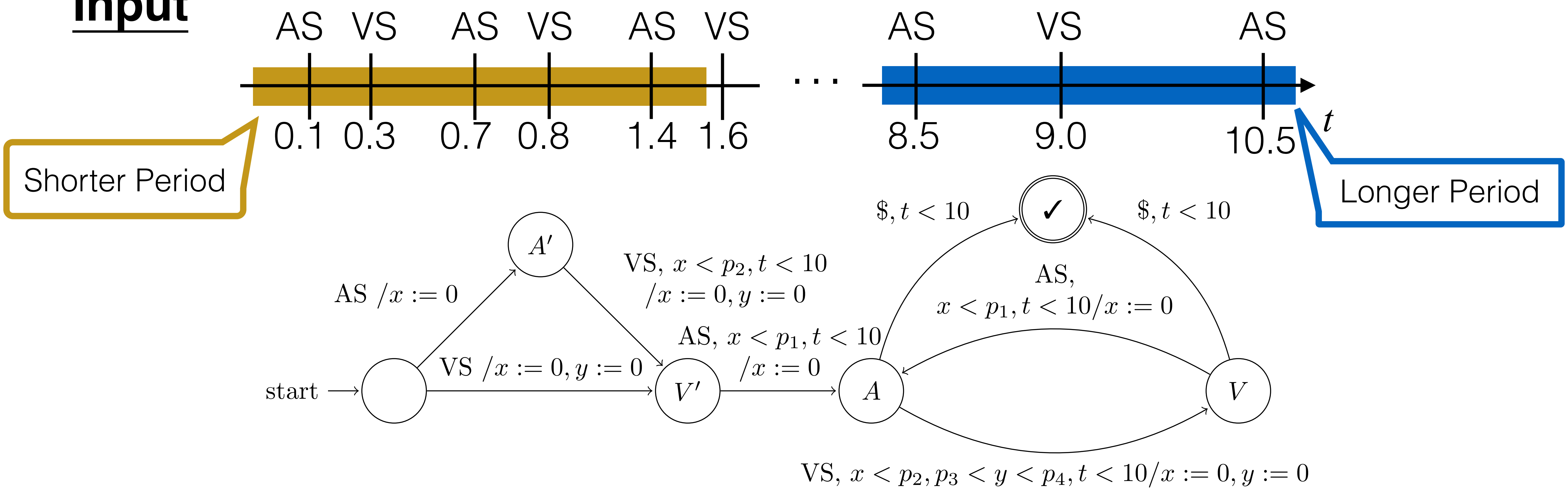
Output

- The intervals **+ param. valuation**, s.t. the ***spec.*** is satisfied in the ***log***
  - e.g., The above behavior occurs in 0.8s-3.4s, $p = 2.5$

PTPM for Periods Detection

Imaginary Example: Electrocardiography (Atrial/Ventricular Spikes)

Input



Output

$$\mathrm{Match}(w, \mathcal{A}) = \boxed{\{(t,t',v) \mid t \in [0,0.1),\ t' \in (1.4,\ 1.6],\ \underline{v(p_1) > \mathbf{0.6}},\ v(p_2) > \mathbf{0.2},\ \ldots\}}$$

$$\cup \ldots \cup \boxed{\{(t,t',v) \mid t \in [1.6,\ 8.5),\ t' \in (10.5,\infty),\ \underline{v(p_1) > \mathbf{1.5}}, v(p_2) > \mathbf{0.5}\}}$$

Contribution

- Give a specialized alg. for parametric timed pattern matching
    - [André, Hasuo, & Waga, ICECCS'18] IMITATOR-based

      Model Checker for PTA
- Optimized the algorithm with **skipping** for string matching (FJS)
- Implementation + experiment
    - we have 3 new Alg. + IMITATOR-based:
      naive, parametric/non-parametric skipping
    - Our algorithms are much faster than IMITATOR-based algorithm

Outline

- Motivation + Introduction
- Technical Part
  - The parametric timed pattern matching problem
  - Naive algorithm for Parametric TPM [Alg. 1]
  - Skipping optimization for Parametric TPM
    - Parametric Skipping Algorithm [Alg. 2]
    - Non-Parametric Skipping Algorithm [Alg. 3]
- Experiment

Parametric Timed Pattern Matching

[André, Hasuo, & Waga, ICECCS'18]

Input

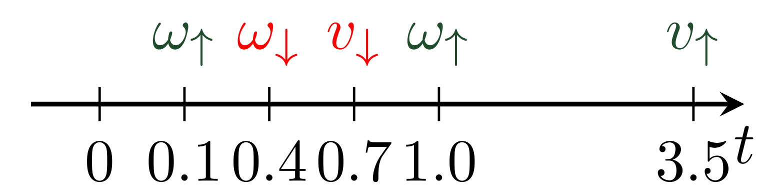
- Timed word $w \in (\Sigma \times \mathbb{R}_{>0})^*$
  - System ***log***
- PTA $\mathcal{A}$
  - Parameterized ***spec.***

Output

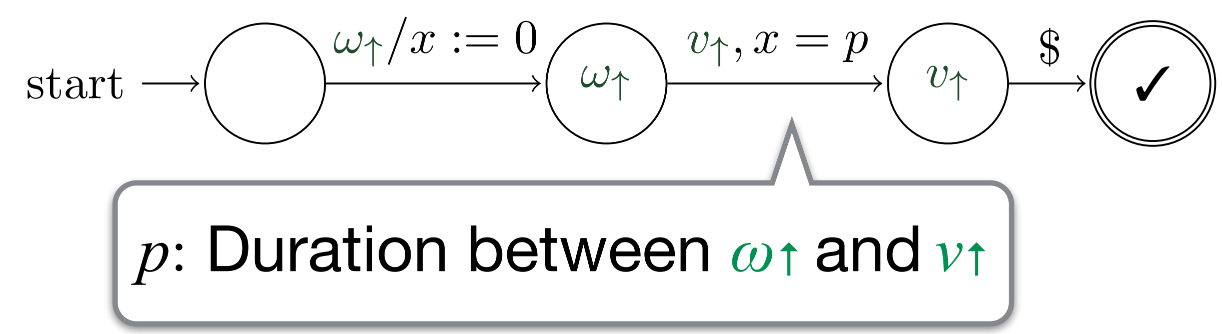
- $\mathrm{Match}(w, \mathcal{A}) = \{(t,t',v) \mid w|_{(t,t')} \in \mathcal{L}(v(\mathcal{A}))\}$
  - Interval (t,t') + param. val. v s.t. spec. is satisfied

Example

$w =$



$\mathcal{A} =$



$$\mathrm{Match}(w, \mathcal{A}) = \{(t,t',v) \mid 0.7 \leq t < 1.0,\ 3.5 < t',\ v(p) = 2.5\}$$

Outline

- Motivation + Introduction
- Technical Part
  - The parametric timed pattern matching problem
  - Naive algorithm for Parametric TPM [Alg. 1]
  - Skipping optimization for Parametric TPM
    - Parametric Skipping Algorithm [Alg. 2]
    - Non-Parametric Skipping Algorithm [Alg. 3]
- Experiment

M. Waga (NII)

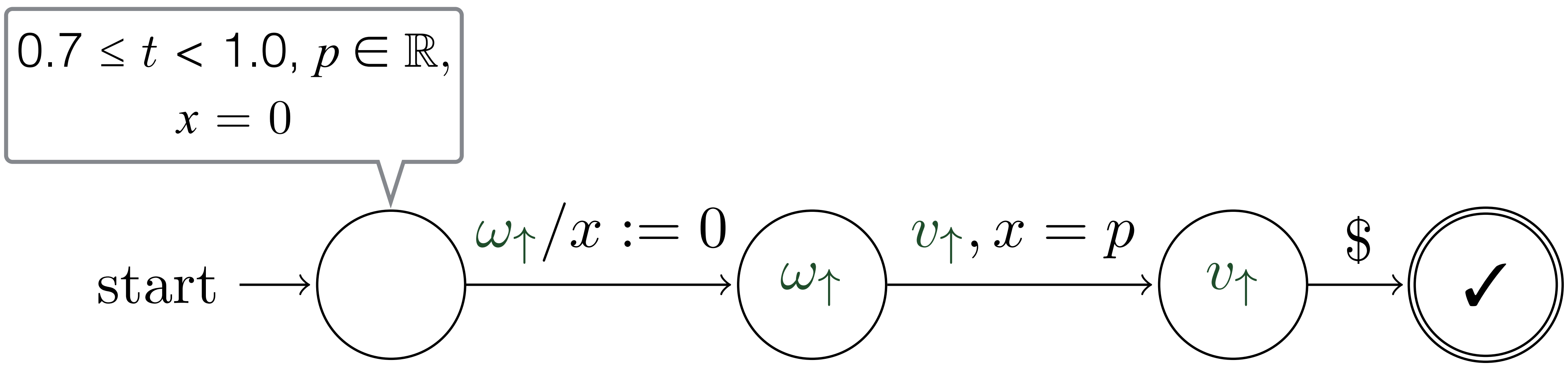
Idea of our (naive) online algorithm

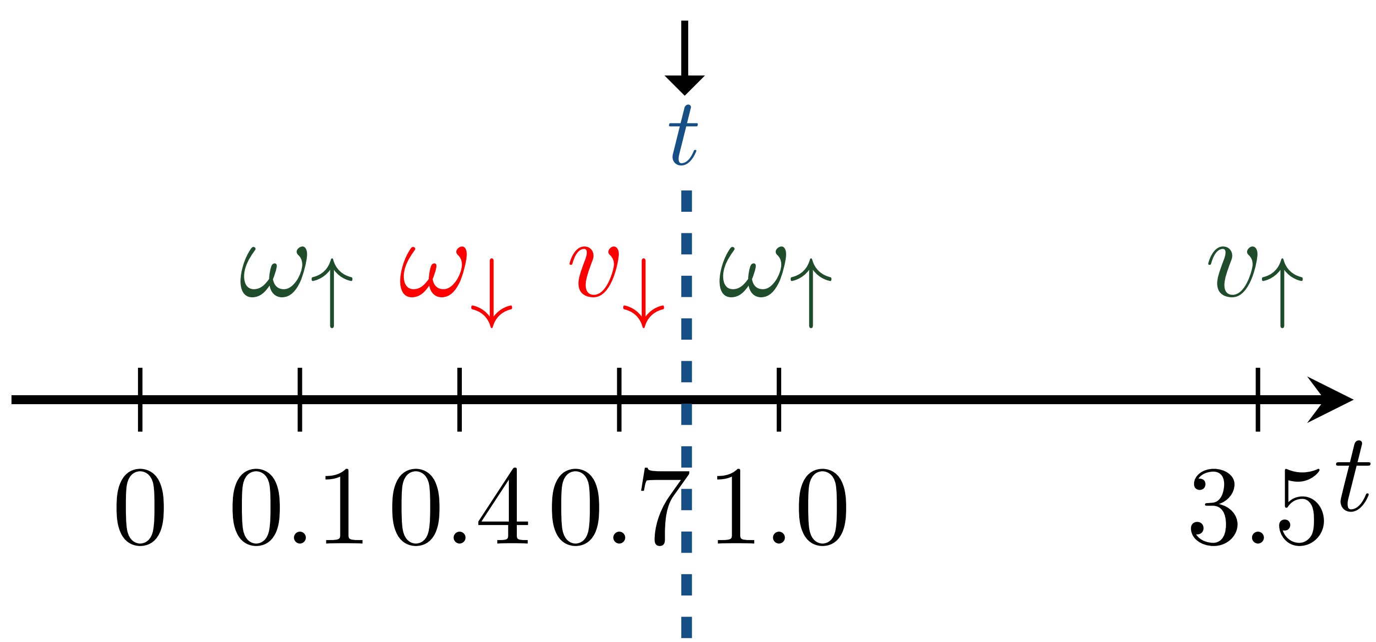
follow the transitions of PTA

+

abstraction of clock/param. val.
by convex polyhedra

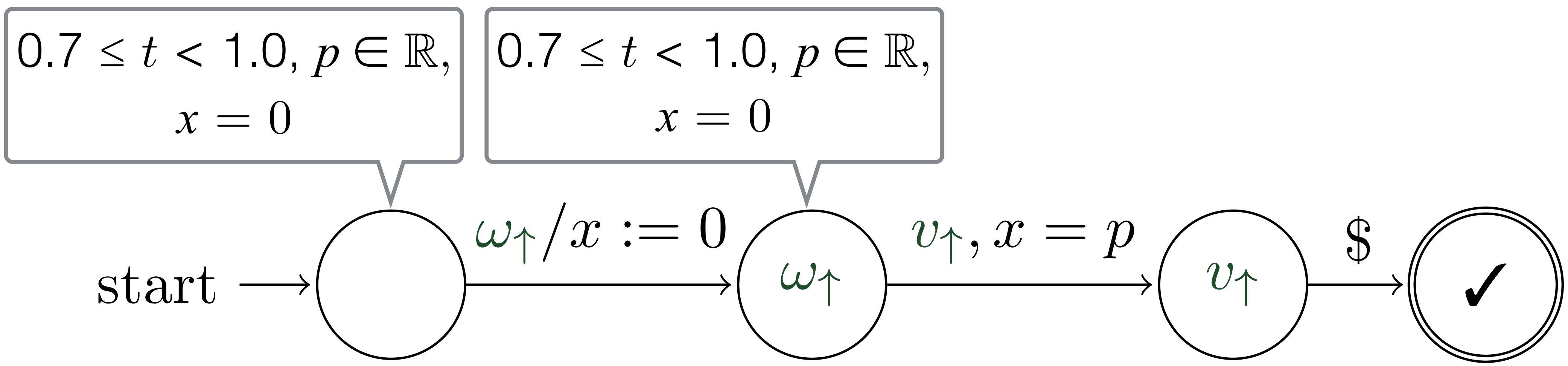
Our online (naive) algorithm

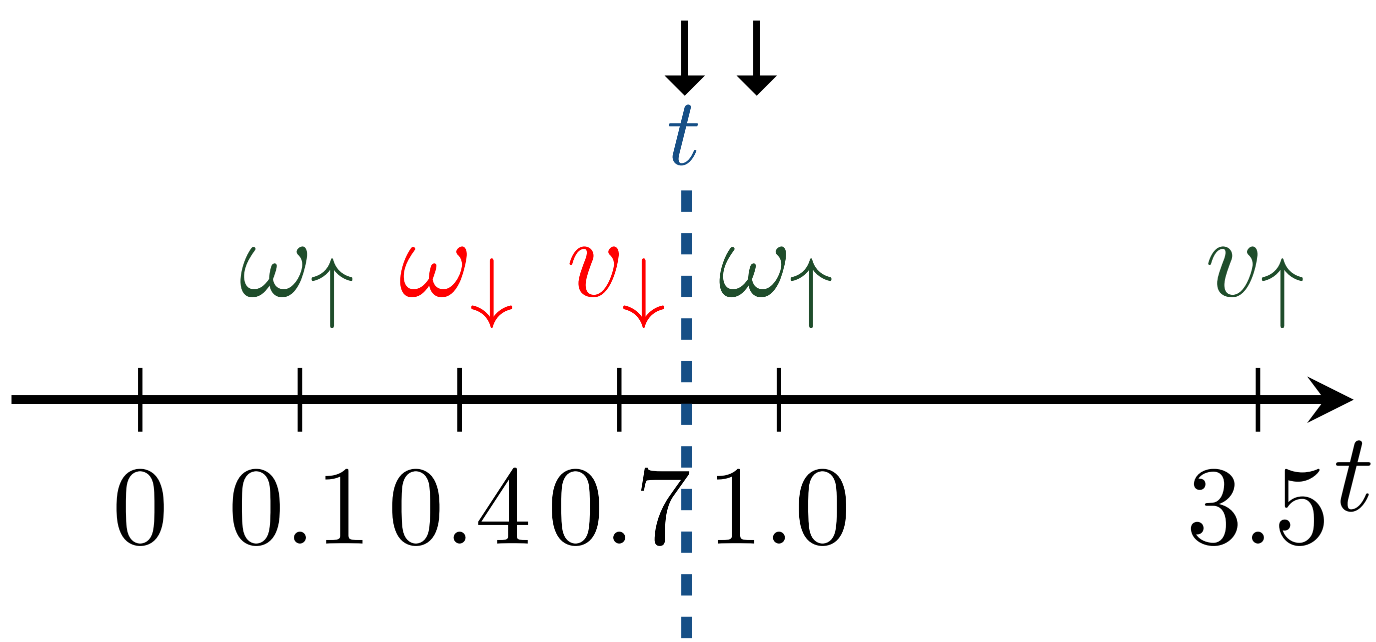




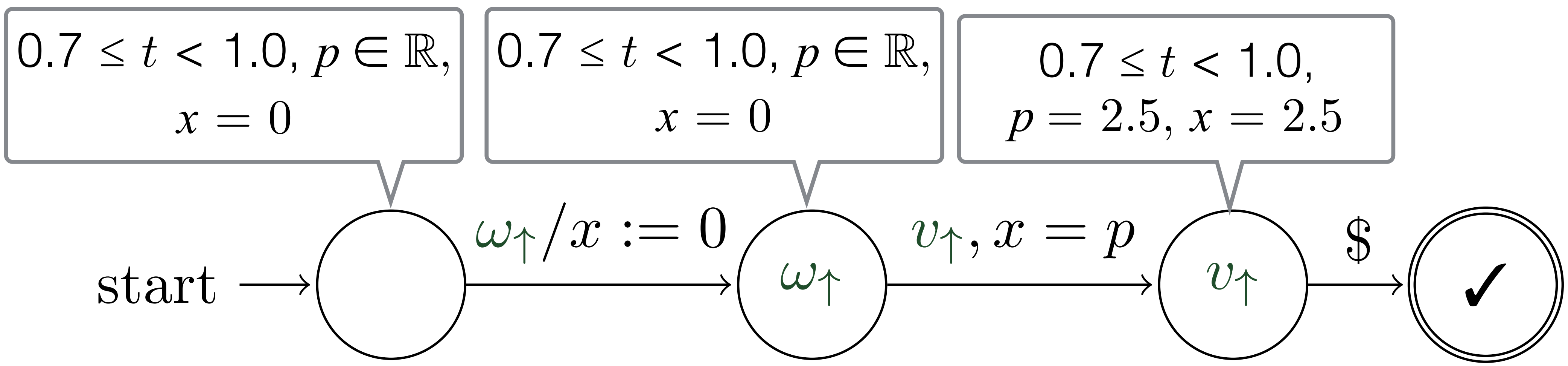
M. Waga (NII)

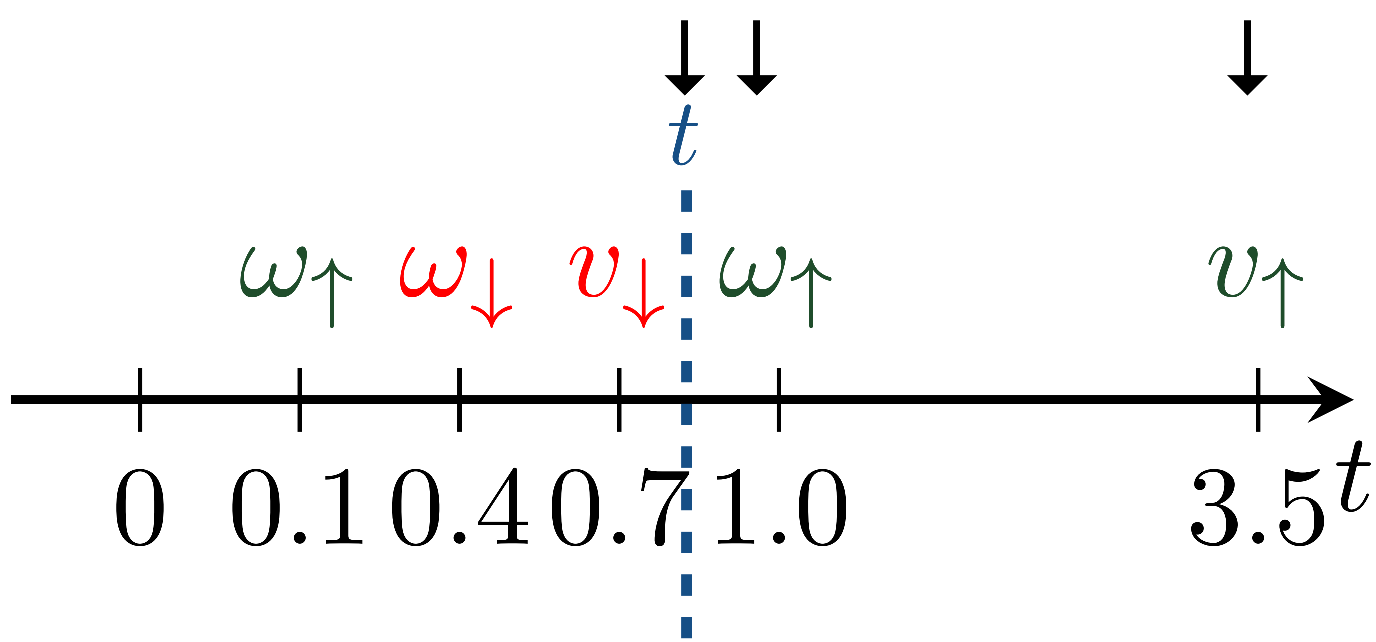
Our online (naive) algorithm





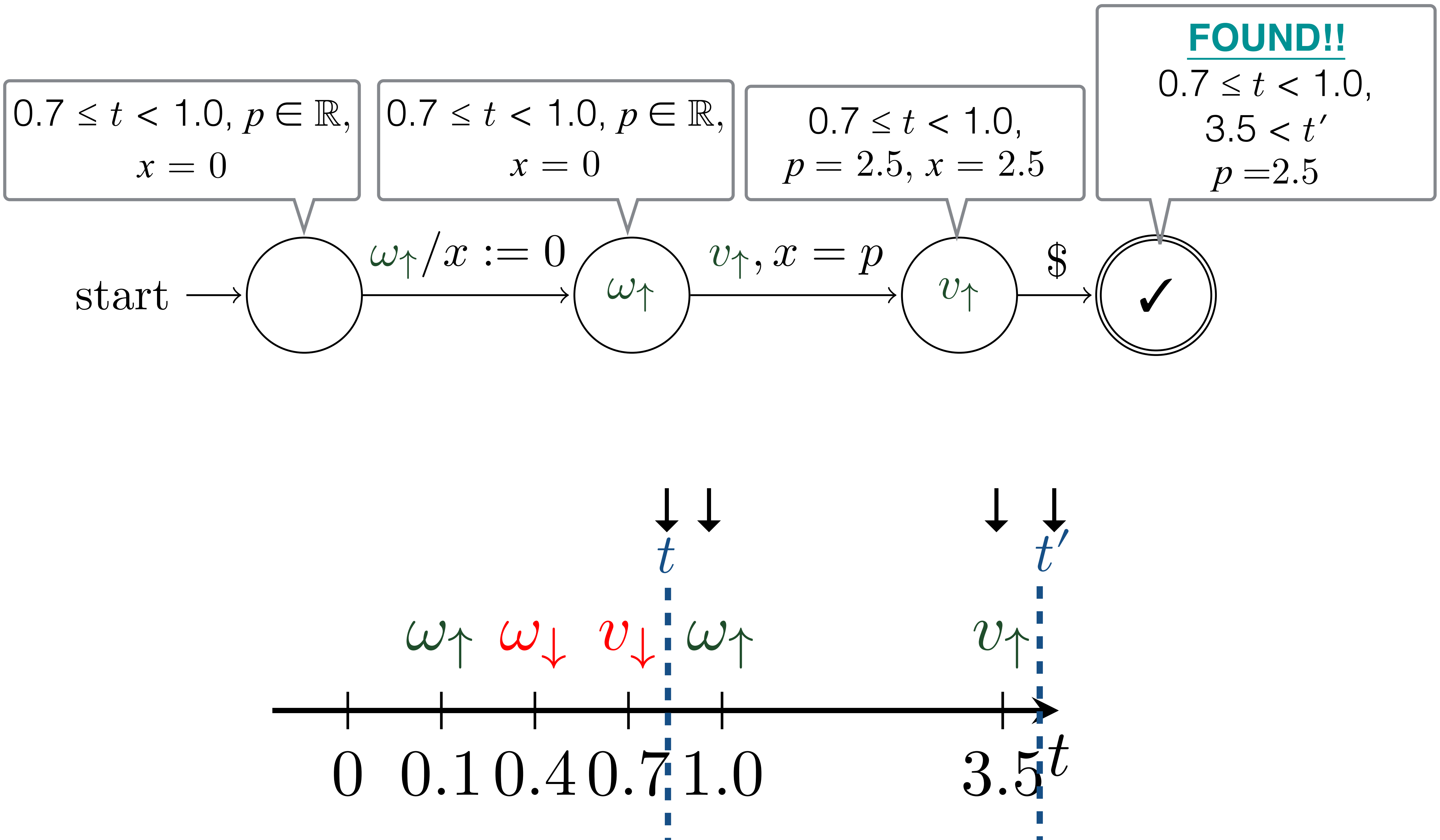
Our online (naive) algorithm



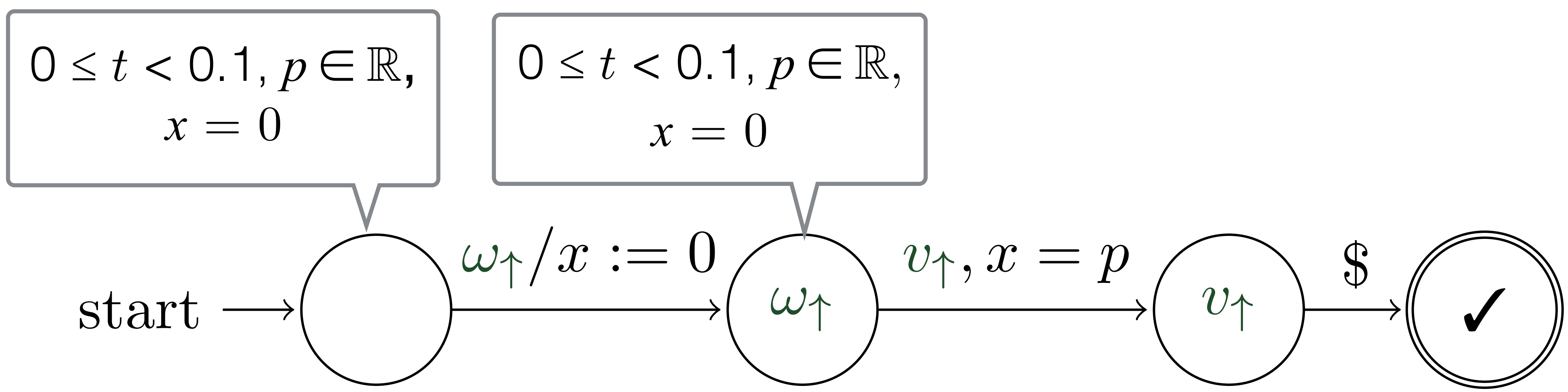


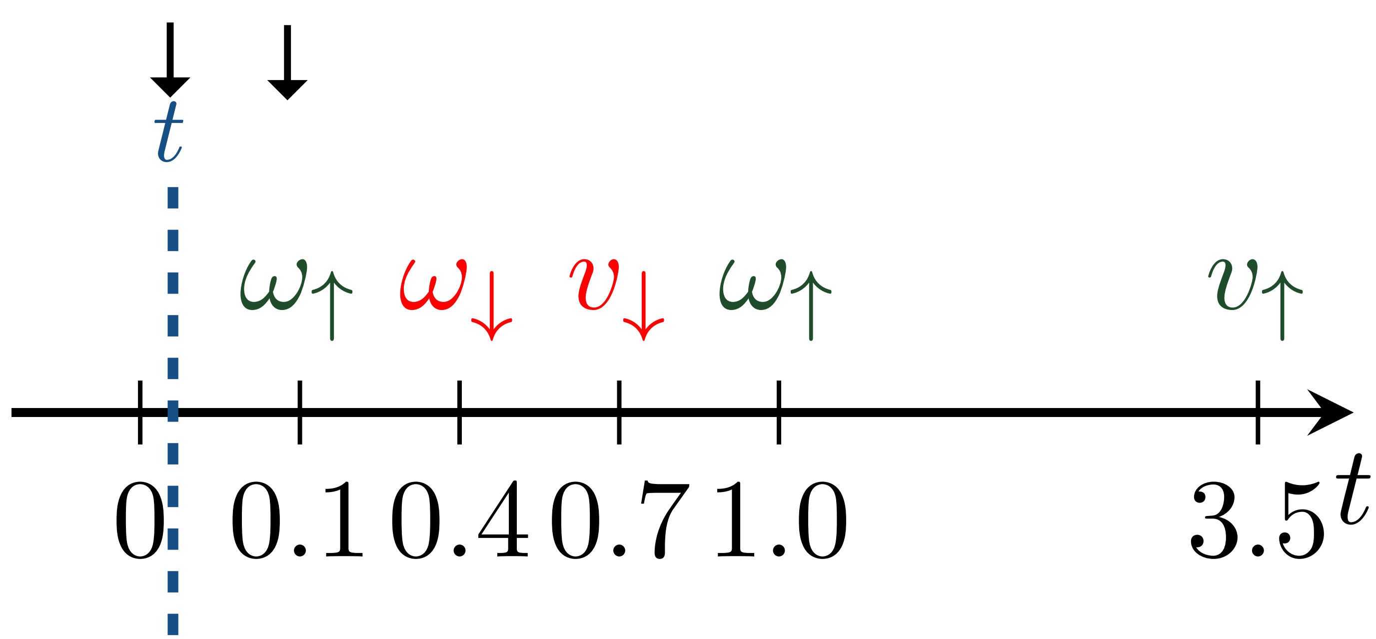
M. Waga (NII)

Our online (naive) algorithm



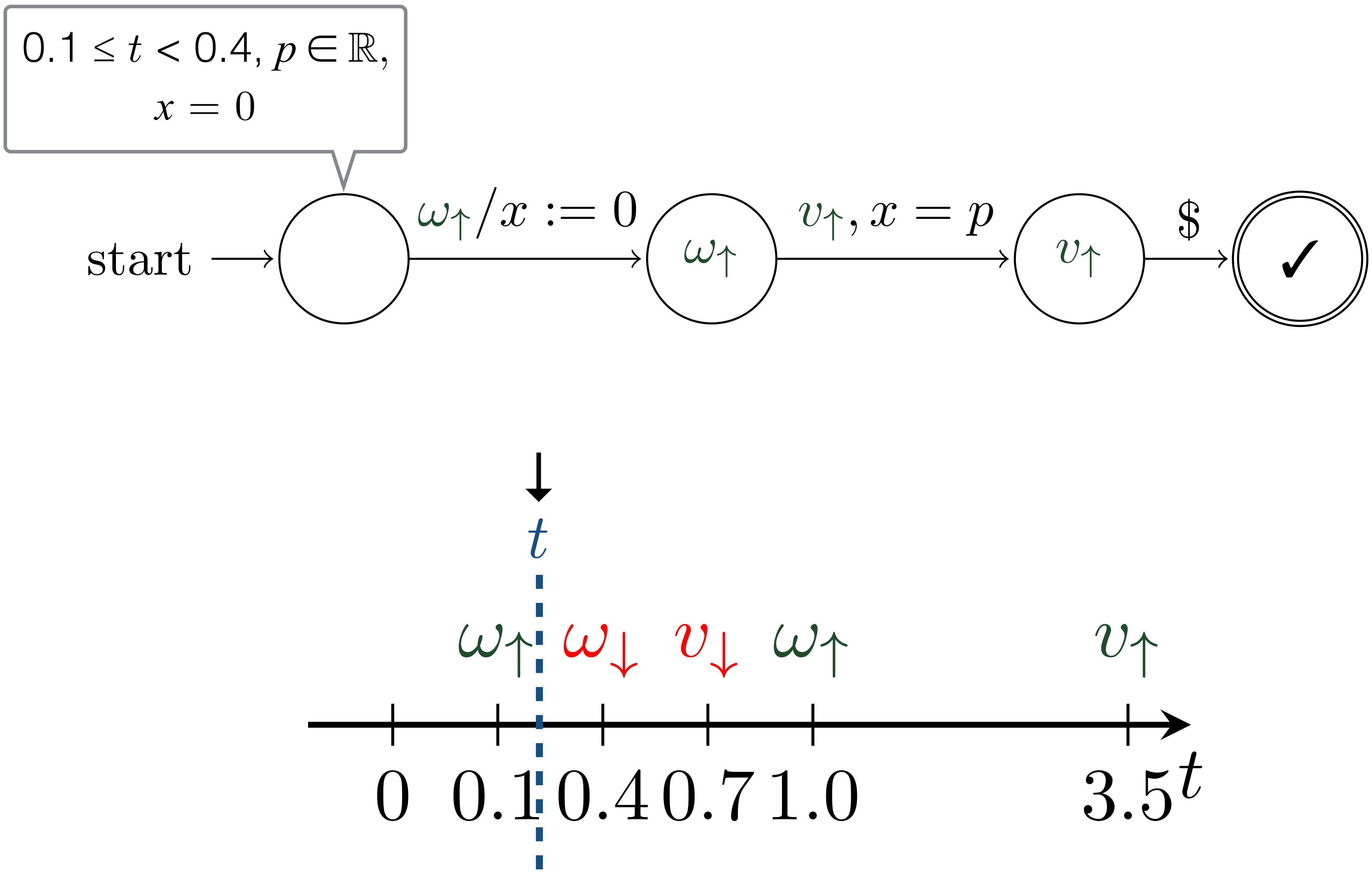
Our online (naive) algorithm





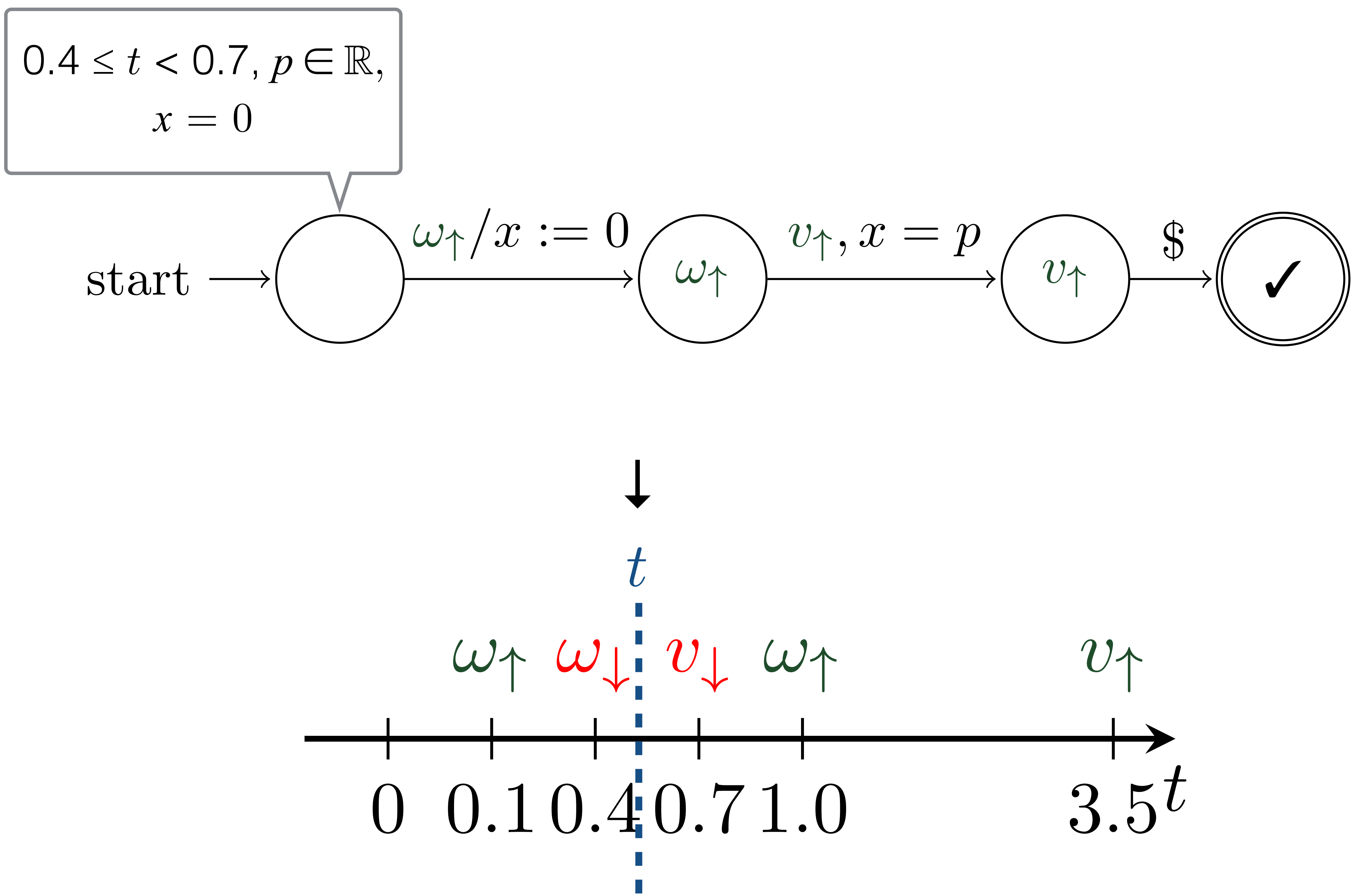
M. Waga (NII)

Our online (naive) algorithm



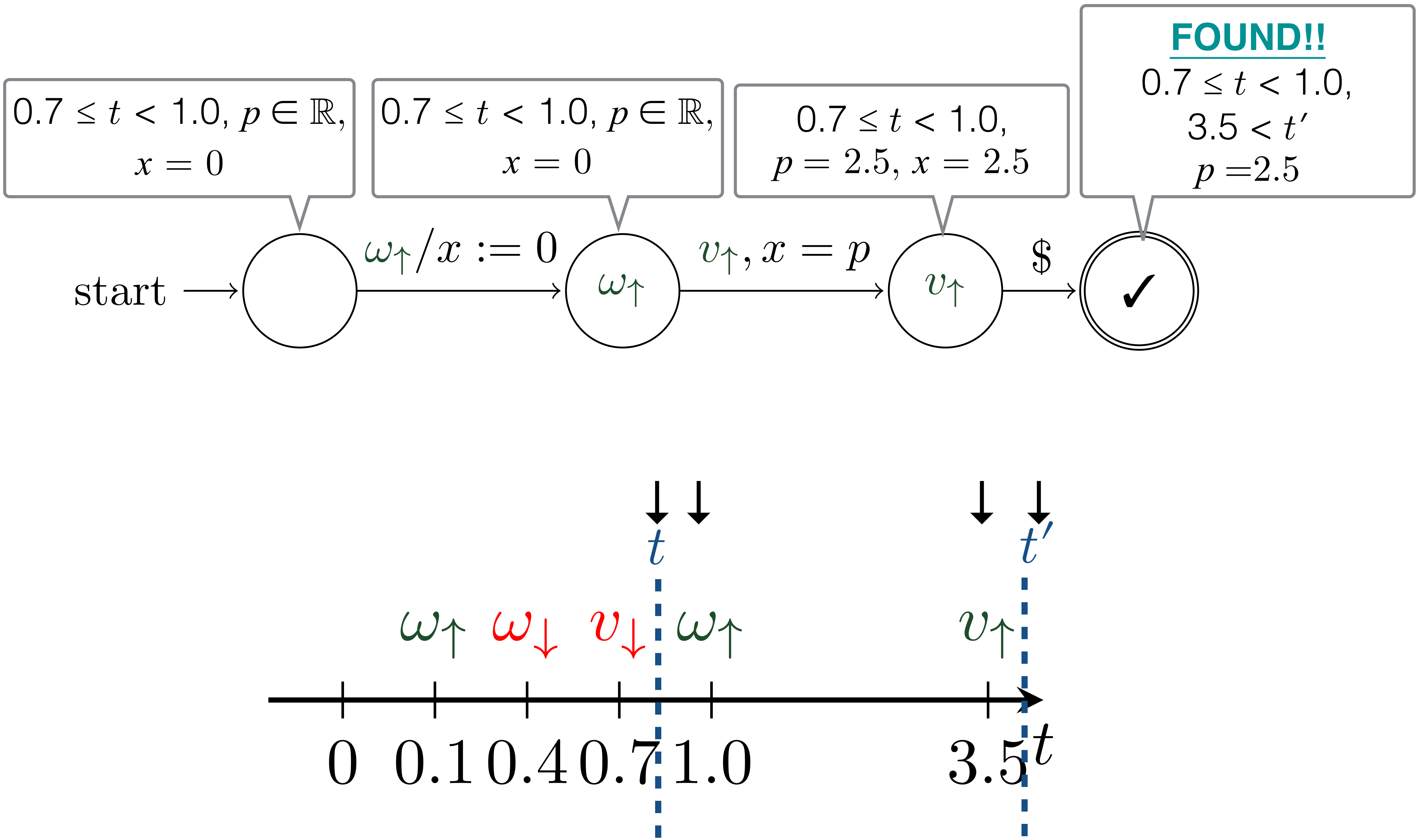
M. Waga (NII)

Our online (naive) algorithm



M. Waga (NII)

Our online (naive) algorithm

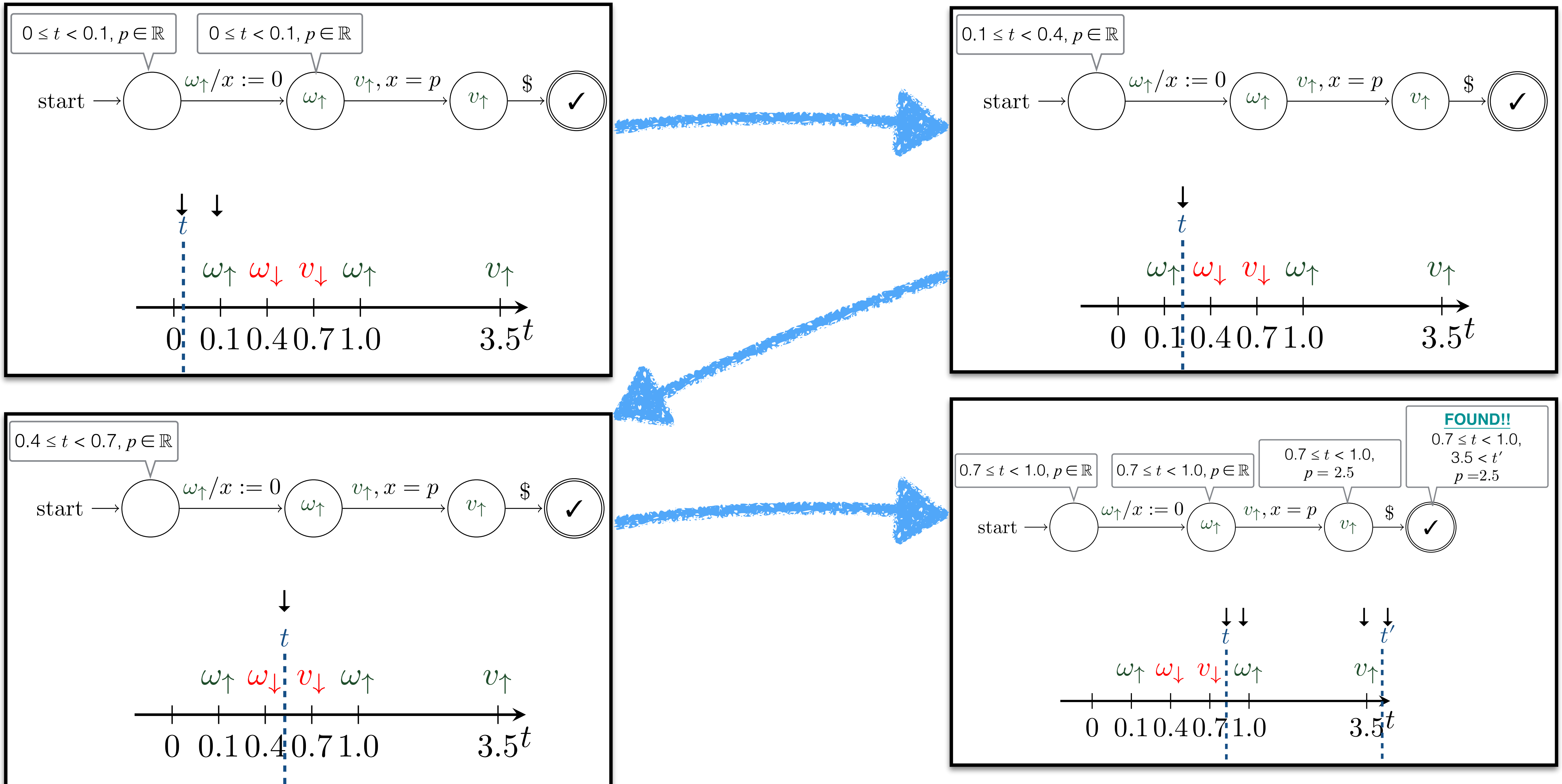


M. Waga (NII)

Outline

- Motivation + Introduction
- Technical Part
  - The parametric timed pattern matching problem
  - Naive algorithm for Parametric TPM [Alg. 1]
  - Skipping optimization for Parametric TPM
    - Parametric Skipping Algorithm [Alg. 2]
    - Non-Parametric Skipping Algorithm [Alg. 3]
- Experiment

Motivation: Skip Unnecessary Trials



Can we skip some of them? ⇒ (Usually) Yes!! by skipping from string matching

M. Waga (NII)

Idea of Skipping in String Matching

- For each length $n \in \mathbb{N}$, check if

$$\boxed{\text{Over-approx. of the read timed word}} \cap \boxed{n\text{-shift + accepted timed words}} = \emptyset$$

  before the matching trials.
- Empty ⇒ no need to try n-shift
- Over-approx. using location $l \in L$ (finite)
  - In PTA, we also use param. val. v: $\mathbb{P} \to \mathbb{R}$
  - Infinite but represented by convex polyhedra

Skipping from Non-Parametric to Parametric

Skipping for TA [Waga+, FORMATS'17]

For each $l \in L$, $n \in \mathbb{N}$, if $\boxed{\mathcal{L}(\mathcal{A}_l) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(\mathcal{A})} = \emptyset$

Over-approx. of the read timed word

n-shift + accepted timed words

(Parametric) Skipping for PTA [Contribution, Alg. 2]

For each $l \in L$, $v: \mathbb{P} \to \mathbb{R}$ $n \in \mathbb{N}$, if there is v': $\mathbb{P} \to \mathbb{R}$ s.t.

$$\boxed{\mathcal{L}(v(\mathcal{A}_l)) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(v'(\mathcal{A}))} = \emptyset$$

Over-approx. of the read timed word for the **given** param. val. v

n-shift + accepted timed words for some param. val. v'

Skipping from Non-Parametric to Parametric

Skipping for TA [Waga+, FORMATS'17]

For each $l \in L$, $n \in \mathbb{N}$, if $\boxed{\mathcal{L}(\mathcal{A}_l) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(\mathcal{A})} = \emptyset$

Over-approx. of the read timed word

n-shift + accepted timed words

(Parametric) Skipping for PTA [Contribution, Alg. 2]

For each $l \in L$, $v \colon \mathbb{P} \to \mathbb{R}$ $n \in \mathbb{N}$, if there is $v' \colon \mathbb{P} \to \mathbb{R}$ s.t.

Runtime Overhead!!

$\boxed{\mathcal{L}(v(\mathcal{A}_l)) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(v'(\mathcal{A}))} = \emptyset$

Over-approx. of the read timed word for the **given** param. val. v

n-shift + accepted timed words for some param. val. v'

Skipping with Less Overhead

(Parametric) Skipping for PTA [Contribution, Alg. 2]

For each $l \in L$, $v\colon \mathbb{P} \to \mathbb{R}$ $n \in \mathbb{N}$, if there is $v'\colon \mathbb{P} \to \mathbb{R}$ s.t.

$$\boxed{\mathcal{L}(v(\mathcal{A}_l)) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(v'(\mathcal{A}))} = \emptyset$$

More overhead
better approx.

Over-approx. of the read timed word for the **given** param. val. v

n-shift + accepted timed words for some param. val. v'

Trade off!!

(Non-Parametric) Skipping for PTA [Contribution, Alg. 3]

For each $l \in L$, $n \in \mathbb{N}$, if there is v, $v'\colon \mathbb{P} \to \mathbb{R}$ s.t.

$$\boxed{\mathcal{L}(v(\mathcal{A}_l)) \cdot \mathcal{T}(\Sigma)} \cap \boxed{\mathcal{T}^n(\Sigma) \cdot \mathcal{L}(v'(\mathcal{A}))} = \emptyset$$

Less overhead
worse approx.

Over-approx. of the read timed word for **some** param. val. v

n-shift + accepted timed words for some param. val. v'

Outline

- Motivation + Introduction
- Technical Part
  - The parametric timed pattern matching problem
  - Naive algorithm for Parametric TPM [Alg. 1]
  - Skipping optimization for Parametric TPM
    - Parametric Skipping Algorithm [Alg. 2]
    - Non-Parametric Skipping Algorithm [Alg. 3]
- Experiment

RQ: Which is the fastest algorithm?

Contribution

[André, Hasuo, & Waga, ICECCS'18]

Algorithms
- Naive
- Parametric Skip
- Non-Parametric Skip
- IMITATOR-based

- Probably, IMITATOR-based is not very fast
  - it solves more general problem (model checking)
- Param. Skip vs. Non-Param. Skip
  - Better over-approx. vs. Less Overhead

Environment of Experiment

- Amazon EC2 c4.large instance
  - 2.9 GHz Intel Xeon E5-2666 v3, 2 vCPUs, 3.75 GiB RAM
- Ubuntu 18.04 LTS (64 bit)
- GCC 7.3.0 with optimization flag -O3
- Used 4 benchmarks
  - **Automotive**: Accel, Gear
  - **Toy**: Blowup, OnlyTiming

Performance for extreme inputs

Comparison with IMITATOR

Table 2: Execution time for GEAR [s]

$\|w\|$	No Skip	Non-Param. Skip	Param. Skip	IMITATOR
1467	0.04	0.05	0.05	1.781
2837	0.0725	0.0805	0.09	3.319
4595	0.124	0.13	0.1405	5.512
5839	0.1585	0.156	0.17	7.132
7301	0.201	0.193	0.2115	8.909
[illegible]	0.241	0.2315	0.2505	10.768
[illegible]	0.2815	0.269	0.2875	12.778
[illegible]	0.322	0.301	0.325	14.724
13183	[illegible]	0.3245	0.353	16.453
14657	0.392	0.361	0.395	18.319

60x faster

Table 3: Execution time for ACCEL [s]

$\|w\|$	No Skip	Non-Param. Skip	Param. Skip	IMITATOR
2559	0.03	0.0515	0.06	2.332
4894	0.0605	0.0605	0.0705	4.663
7799	0.1005	0.071	0.08	7.532
10045	0.13	0.08	0.09	9.731
12531	0.161	0.09	0.1	12.503
15375	0.1985	0.1005	0.113	15.583
17688	0.2265	0.1095	0.1215	17.754
20299	0.261	0.115	0.1325	[illegible]
22691	0.288	0.121	[illegible]	23.044
25137	0.3205	0.1315	0.159	25.815

200x faster

Table 4: Execution time for BLOWUP [s]

$\|w\|$	No Skip	Non-Param. Skip	Param. Skip	IMITATOR
2000	66.75	68.0125	67.9735	OutOfMemory
4000	267.795	271.642	269.084	OutOfMemory
6000	601.335	611.782	607.58	OutOfMemory
8000	1081.42	1081.25	1079	OutOfMemory
10000	1678.15	1688.22	1694.53	OutOfMemory

out of memory!!

Table 5: Execution time for ONLYTIMING [s]

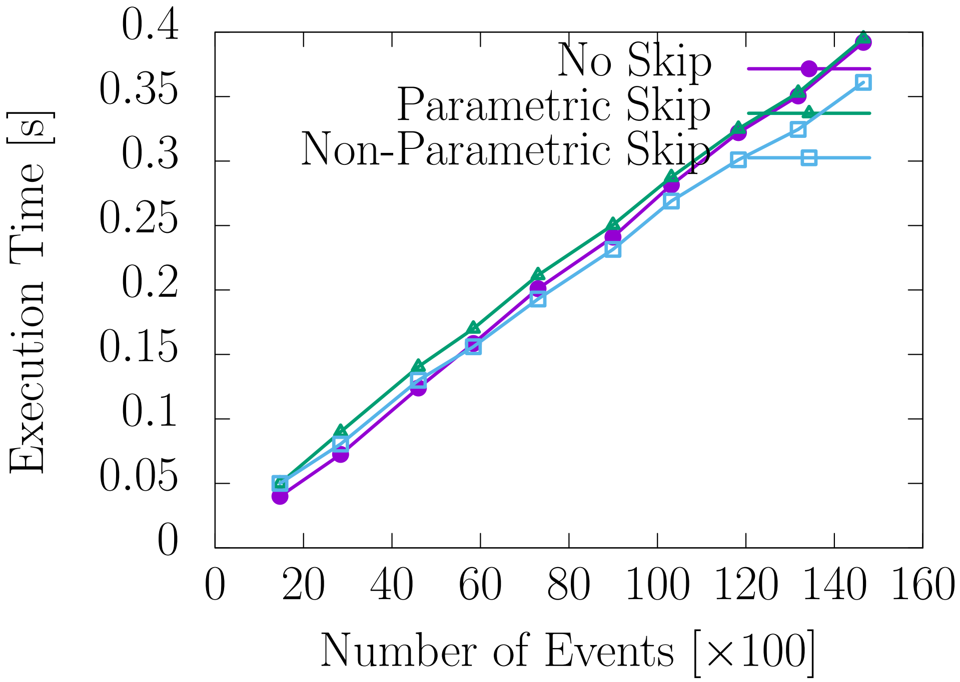
$\|w\|$	No Skip	Non-Param. Skip	Param. Skip	IMITATOR
1000	0.0995	0.1305	0.11	1.690
2000	0.191	0.23	0.191	3.518
3000	0.2905	0.3265	0.273	5.499
4000	0.3905	0.426	0.3525	7.396
5000	0.488	0.5225	0.4325	9.123
6000	0.588	0.6235	0.517	11.005

20x faster

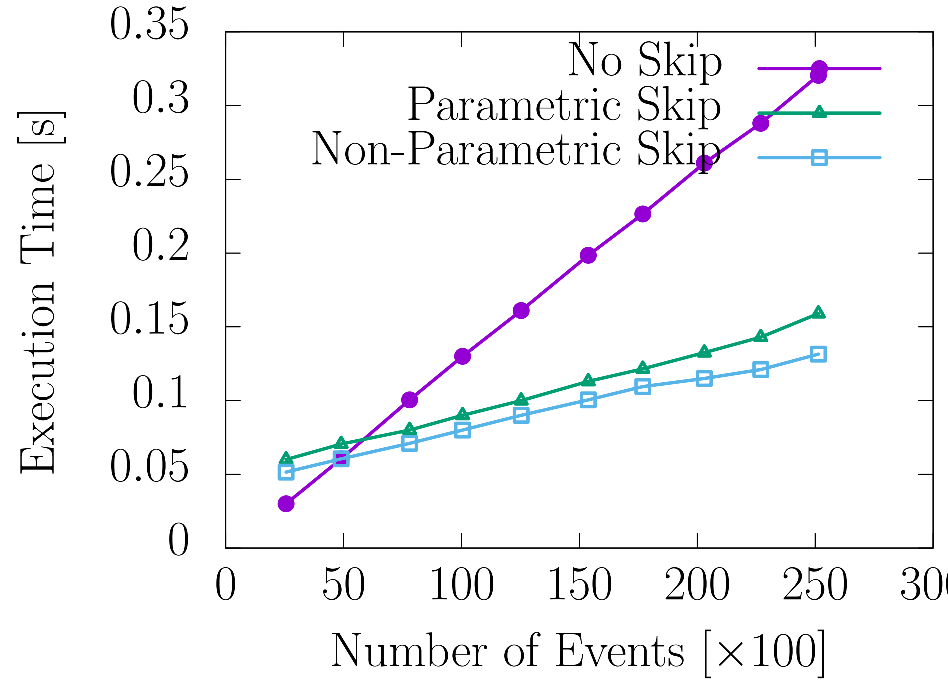
M. Waga (NII)

Comparison among ours (Accel & Gear)

Accel



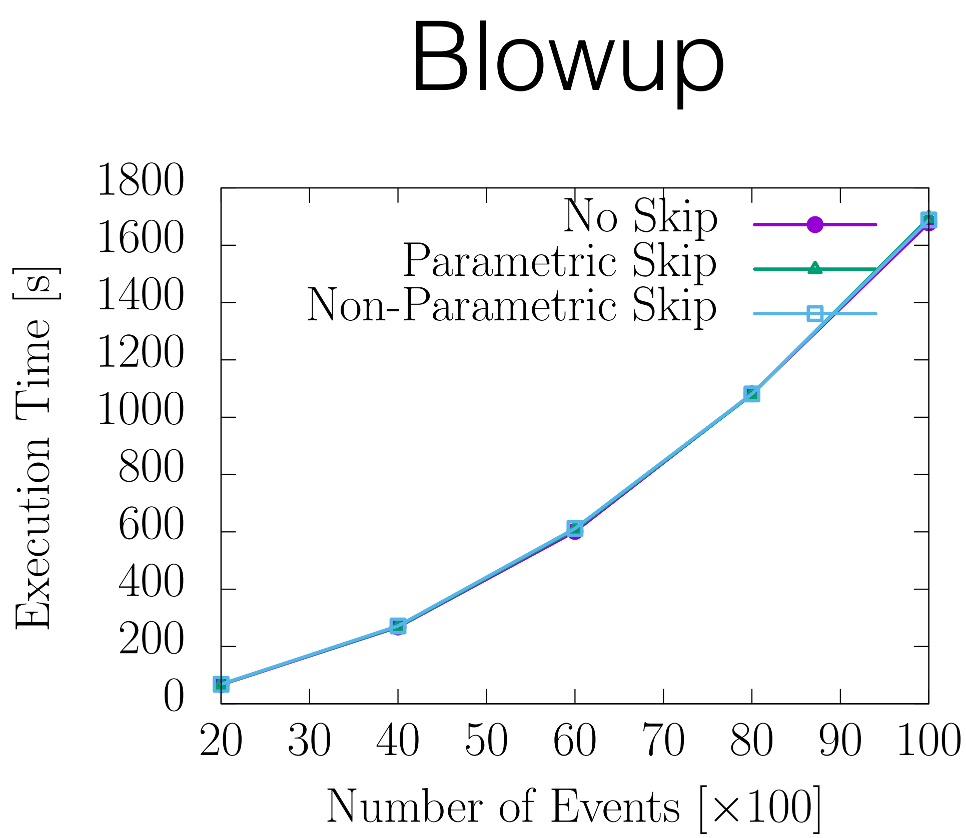
Gear

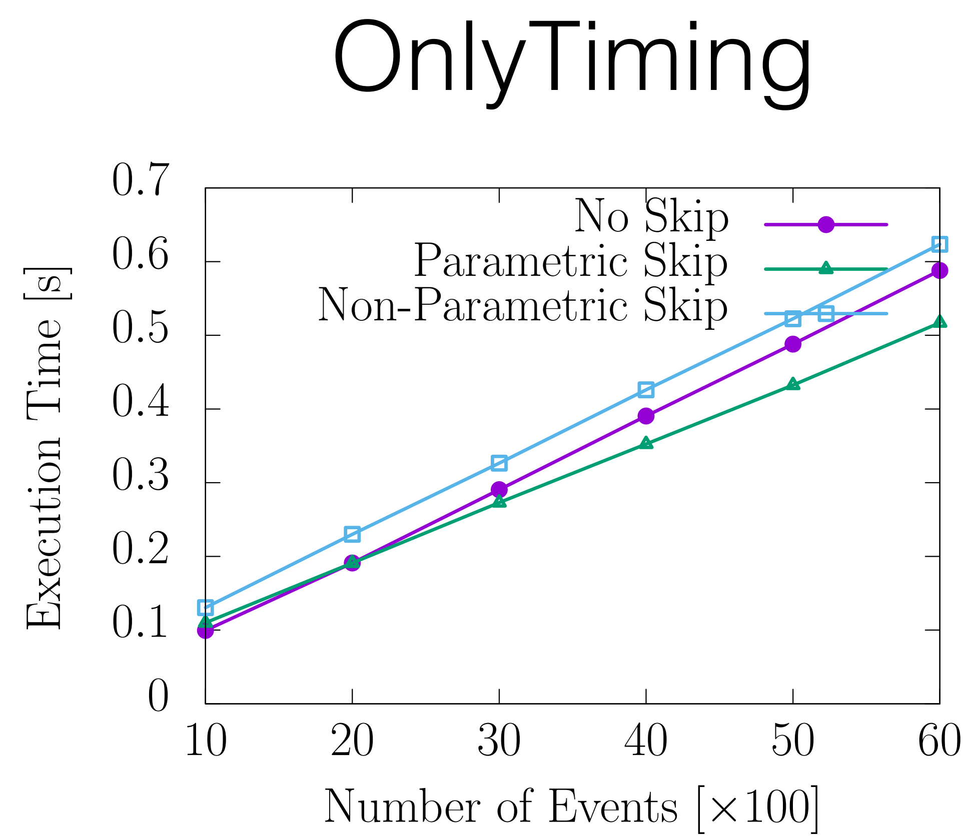


- No Skip has the steepest slope ⇒ worst scalability
- Parametric Skip is slower than Non-Parametric Skip due to the overhead

M. Waga (NII)

Comparison among ours (Blowup & OnlyTiming)





- Blowup: Skipping does not help much
  - Exponential blowup vs. constant speed up by skipping
- OnlyTiming: No Skip has the steepest slope ⇒ worst scalability

  Parametric Skip is the fastest due to the better over-approx.

Conclusion

- Give a specialized alg. for param. timed pattern matching
- Optimized the algorithm by **skipping** from string matching
- Our algorithms are much faster than the state-of-the-art (IMITATOR-based algorithm)
- Param. vs. Non-Param. Skipping depends on the Autom.

Future Works

- Hybrid of Parametric/Non-Parametric Skipping
  - Maybe the best trade-off
- More expressive logic (e.g., FOL)
- Case study
  - not only automotive domain but also medical CPS or IoT (security)

Appendix

Why Autom? not TL or RE?

Cons.

- Difficult to write (and read?) for the end user

Pros.

- More straightforward online monitoring algorithm
- Optimization technique from untimed to timed
- TRE → TA, MITL → TA are possible
  - TA as common platform
  - In our industrial collaboration, we use TRE → TA

e.g., **Skipping** in this talk

Easy 😃

Not easy…😟