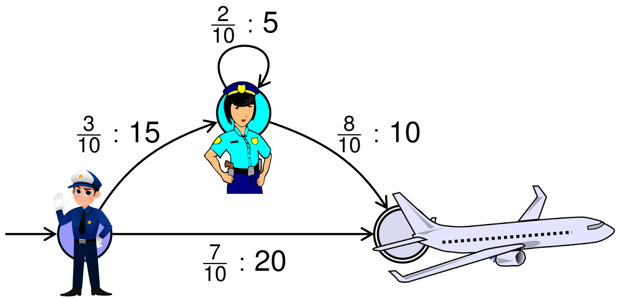
Counting Problems for Parikh Images

Christoph Haase *Stefan Kiefer* Markus Lohrey

MFCS 2017, Aalborg
25 August 2017

The Cost Problem



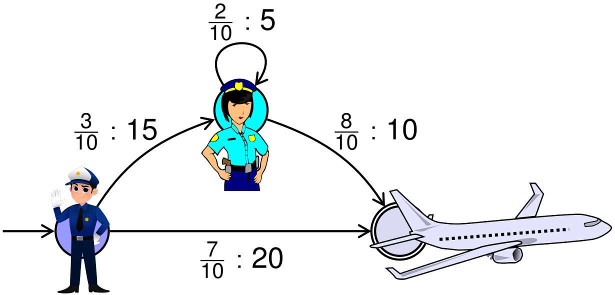
What is the probability to reach the gate in 25–45min?
Quantiles?

Cost Problem :=

Input: Cost Markov chain
cost formula φ ← $25 \leq \text{cost} \leq 45$

Output: $\Pr(\varphi)$

The Cost Problem



What is the probability to reach the gate in 25–45min?
Quantiles?

Cost Problem :=

Input: Cost Markov chain
cost formula φ ← $25 \leq \text{cost} \leq 45$
threshold $\tau \in [0, 1]$

Output: Is $\Pr(\varphi) \geq \tau$?

Complexity of the Cost Problem

Theorem (Laroussinie, Sproston, FoSSaCS'05)

The cost problem is in EXPTIME.
The cost problem is NP-hard.

Complexity of the Cost Problem

Theorem (Laroussinie, Sproston, FoSSaCS'05)

The cost problem is in EXPTIME.

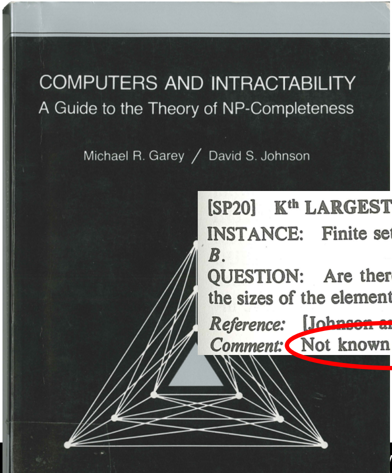
The cost problem is NP-hard. ← by reduction from the *K*th largest subset problem

Complexity of the Cost Problem

Theorem (Laroussinie, Sproston, FoSSaCS'05)

The cost problem is in EXPTIME.

The cost problem is NP-hard. ← by reduction from the *K*th largest subset problem



[SP20] **K^th LARGEST SUBSET (*)**

INSTANCE: Finite set A, size $s(a) \in Z^+$ for each $a \in A$, positive integers K and B.

QUESTION: Are there K or more distinct subsets $A' \subseteq A$ for which the sum of the sizes of the elements in A' does not exceed B?

Reference: [Johnson and Kashdan, 1976]. Transformation from SUBSET SUM.

Comment: Not known to be in NP. Solvable in pseudo-polynomial time (polyno-

Complexity of the Cost Problem

Theorem (Laroussinie, Sproston, FoSSaCS'05)

The cost problem is in EXPTIME.
The cost problem is NP-hard. ← by reduction from the Kth largest subset problem

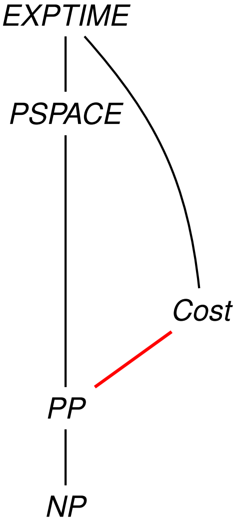
Theorem (HK, IPL'16)

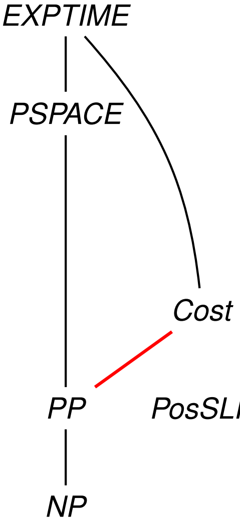
The Kth largest subset problem is PP-complete. ← PP: a superset of NP

Corollary

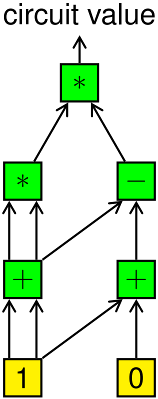
The cost problem is PP-hard.

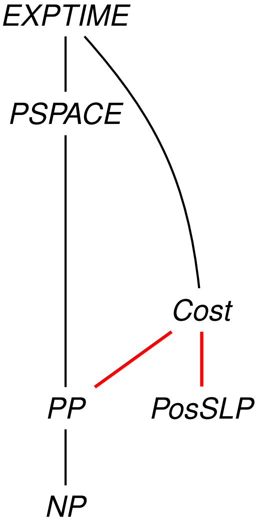
Complexity of the Cost Problem





PosSLP := Input: arithmetic circuit
Output: Is the value > 0 ?



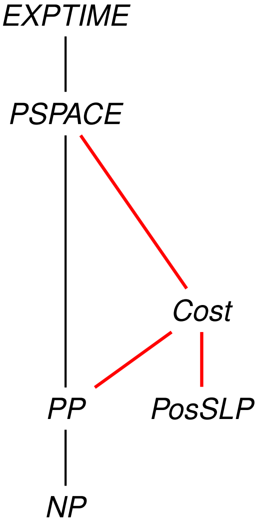


Theorem (HK, ICALP'15)

The cost problem is PosSLP-hard.

PosSLP :=
Input: arithmetic circuit
Output: Is the value > 0 ?

Complexity of the Cost Problem

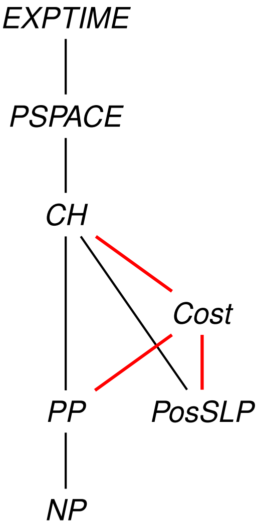


Theorem (HK, ICALP'15)

The cost problem is PosSLP-hard.
The cost problem is in PSPACE.

PosSLP $:=$ Input: arithmetic circuit
Output: Is the value > 0 ?

Complexity of the Cost Problem



Theorem (HK, ICALP'15)

The cost problem is PosSLP-hard.
The cost problem is in PSPACE.

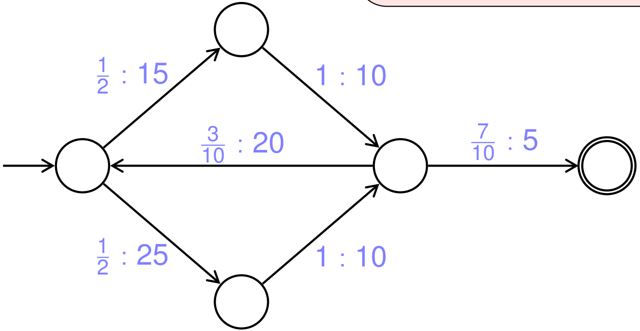
Theorem (HKL, LICS'17)

The cost problem is in CH.

Solving the Cost Problem

Cost Problem :=

Input:	Cost Markov chain cost formula φ
Output:	$\Pr(\varphi)$

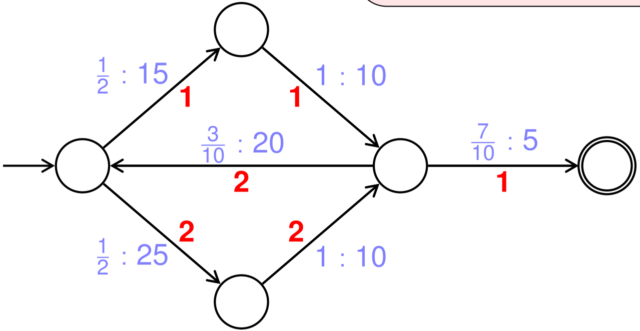


Solving the Cost Problem

Cost Problem :=

Input: Cost Markov chain
cost formula φ

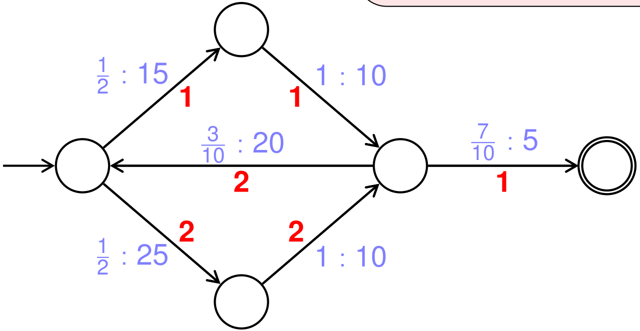
Output: $\Pr(\varphi)$



Solving the Cost Problem

Cost Problem :=

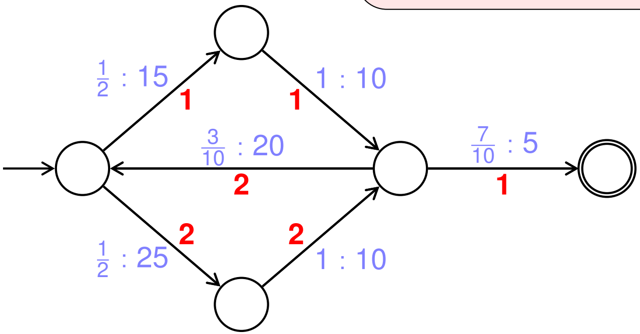
Input:	Cost Markov chain
	cost formula φ
Output:	$\Pr(\varphi)$



Enumerate the **Parikh images**.

Solving the Cost Problem

Cost Problem := Input: Cost Markov chain
cost formula φ
Output: $\Pr(\varphi)$

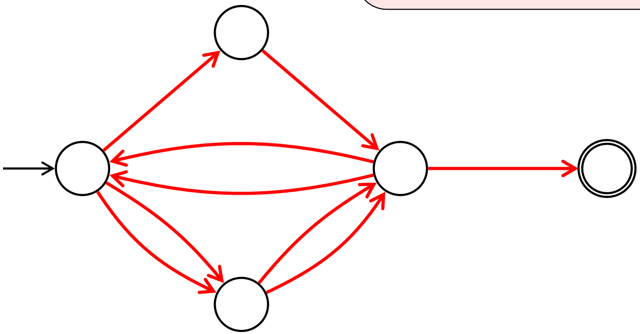


Enumerate the **Parikh images**.
Problem: there might be multiple paths per Parikh image.

Solving the Cost Problem

Cost Problem :=

Input:	Cost Markov chain
	cost formula φ
Output:	$\Pr(\varphi)$

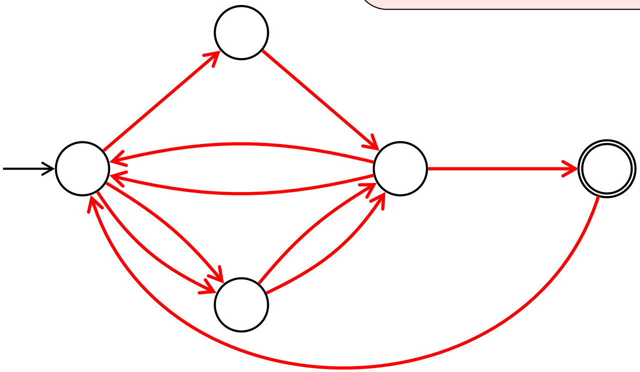


Enumerate the **Parikh images**.
Problem: there might be multiple paths per Parikh image.

Solving the Cost Problem

Cost Problem :=

Input:	Cost Markov chain
	cost formula φ
Output:	$\Pr(\varphi)$



Enumerate the **Parikh images**.

Problem: there might be multiple paths per Parikh image.

The BEST Theorem

Theorem (de Bruijn, van Aardenne-Ehrenfest, Smith, Tutte)

The number of Eulerian cycles in an Eulerian graph G equals

$$t(G) \cdot \prod_{v \in V} (d(v) - 1)!$$

number of directed spanning trees in G

Theorem (Tutte's matrix-tree theorem)

$$t(G) = \det(L(G)_{11})$$

Laplacian of G

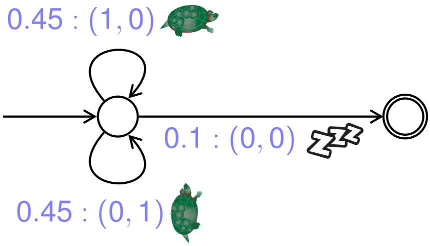
QUANT: A Tool for the Cost Problem

Theorem ([HK, ICALP'15], [HK, IPL'16], [HKL, LICS'17])

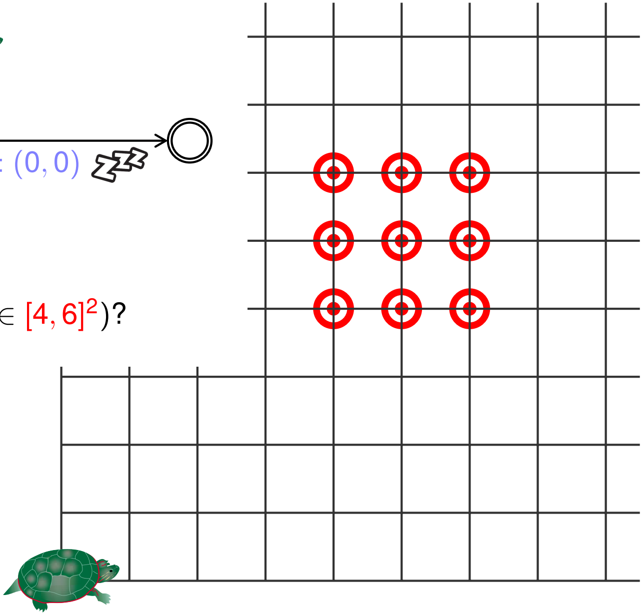
The cost problem is hard for PP and PosSLP.
The cost problem is in CH.

```
while solver.check() == sat:
    model = solver.model()

    for t in T:
        v = model[tv[t]].as_long()
        M.itemset((t[0], t[1]), v)

    minor = Decimal(round(np.linalg.det(laplacian(M)[1:,1:])))
    degs = Decimal(np.prod(map(math.factorial, degreeVector(M) -
                                   np.ones(m+1))))
    weights = Decimal(np.prod(map(math.factorial, [ model[tv[t]].as_long()
                                                 for t in T ])))
    probs = Decimal(np.prod([math.pow(P[t],
                                       model[tv[t]].as_long()) for t in T]))
    p += ((minor * degs) / weights) * probs

    if p > threshold:
        return p

    solver.add(Or([tv[t] != model[tv[t]] for t in T]))
return p
```

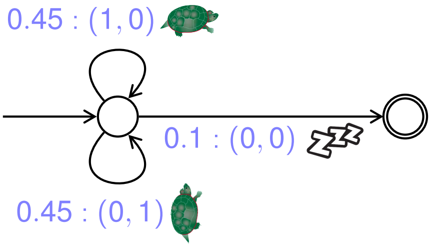
QUANT: A Tool for the Cost Problem



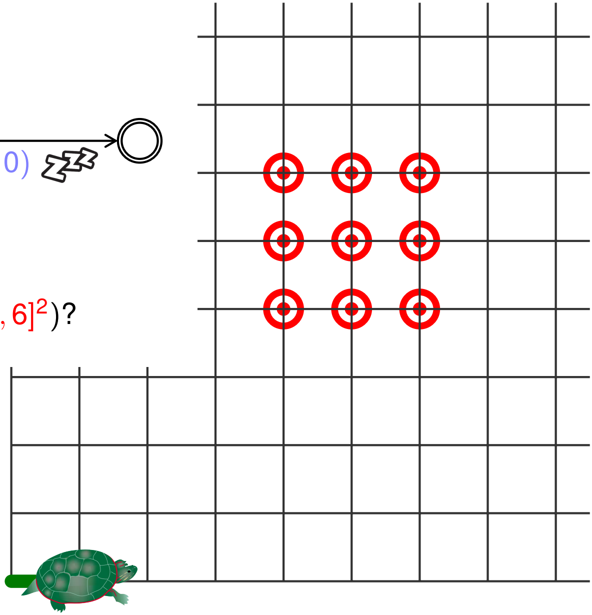
What is Pr(cost $\in [4, 6]^2$)?



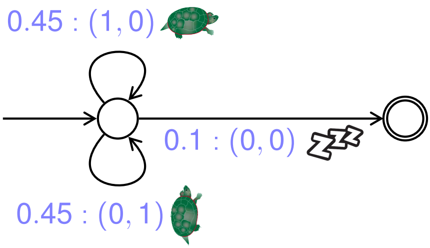
QUANT: A Tool for the Cost Problem



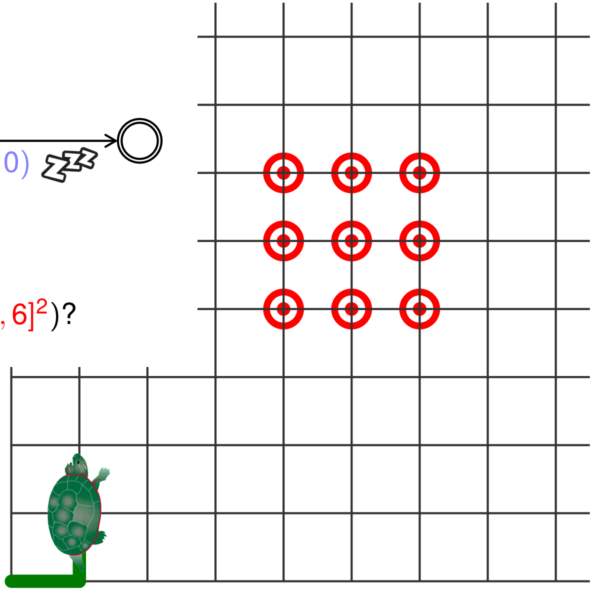
What is Pr(cost $\in [4, 6]^2$)?



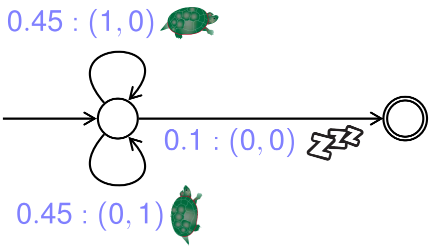
QUANT: A Tool for the Cost Problem



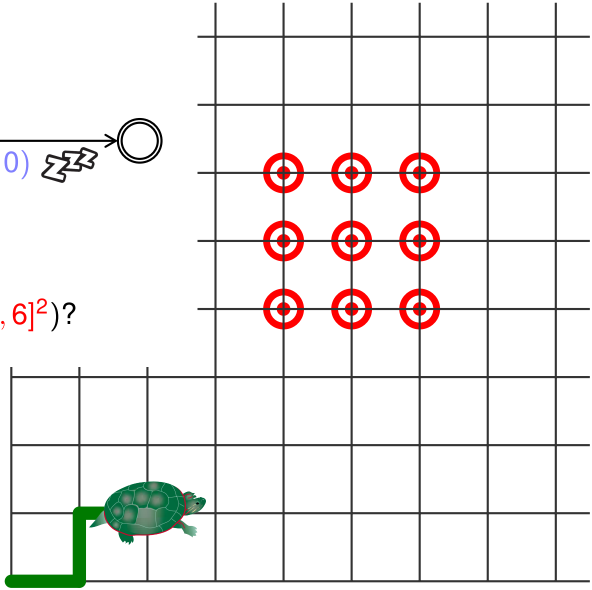
What is Pr(cost $\in [4, 6]^2$)?



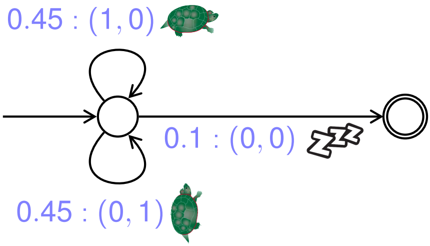
QUANT: A Tool for the Cost Problem



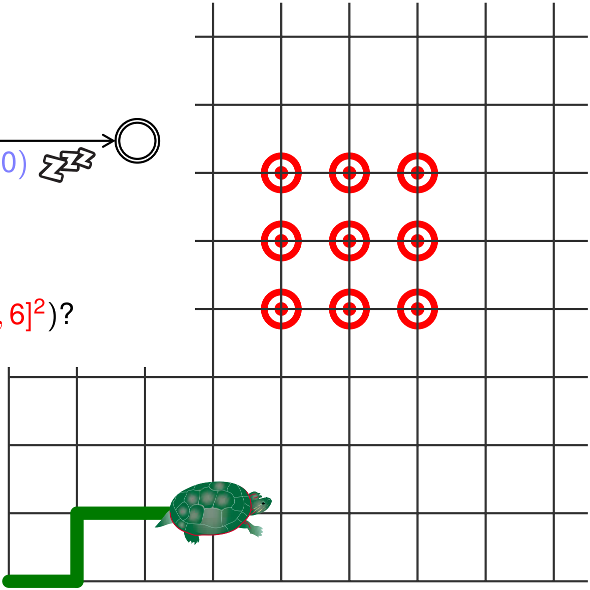
What is Pr(cost $\in [4, 6]^2$)?

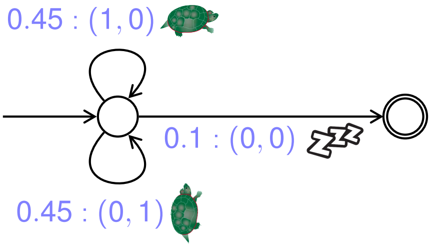


QUANT: A Tool for the Cost Problem

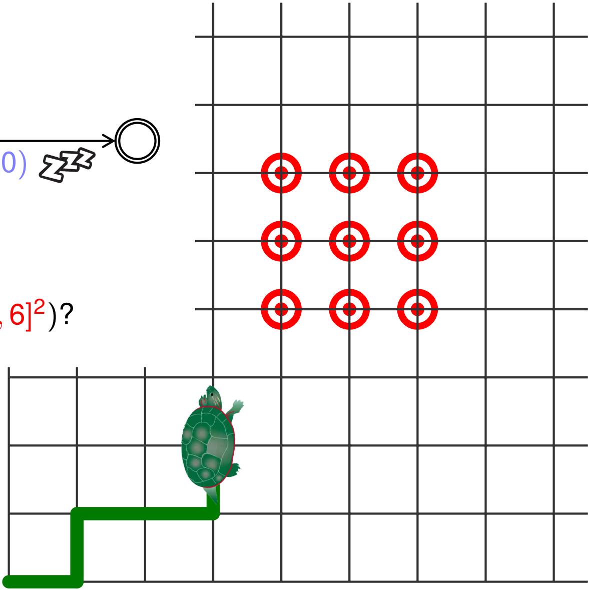


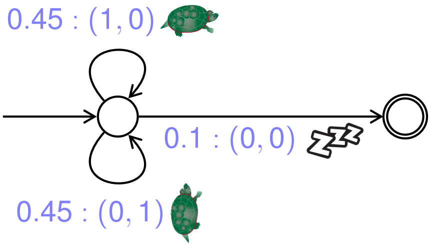
What is $\Pr(\text{cost} \in [4, 6]^2)$?



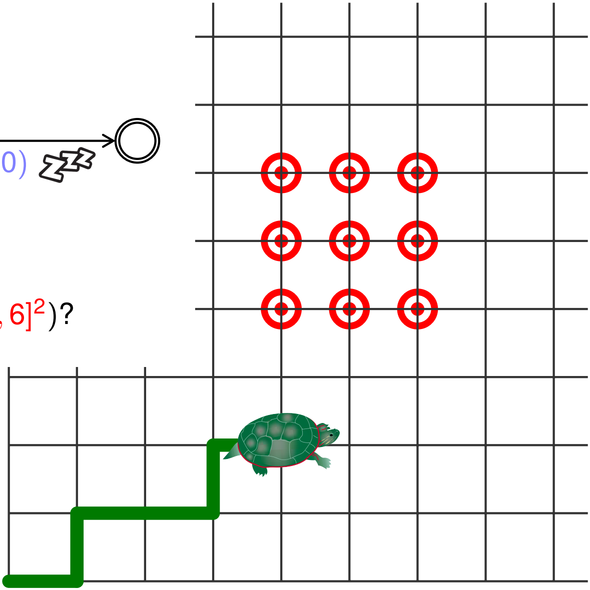


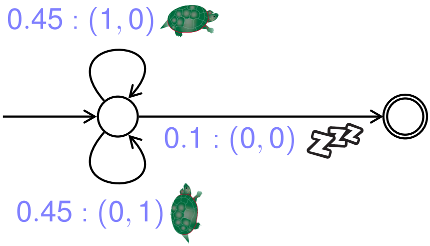
What is Pr(cost $\in [4, 6]^2$)?



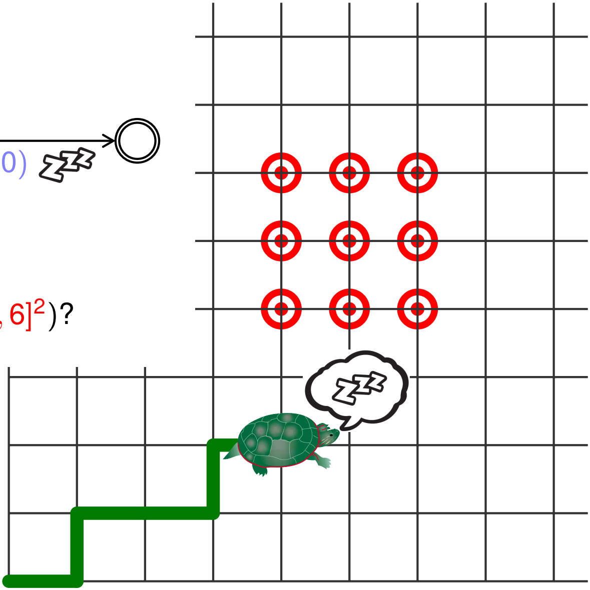


What is $\Pr(\text{cost} \in [4, 6]^2)$?

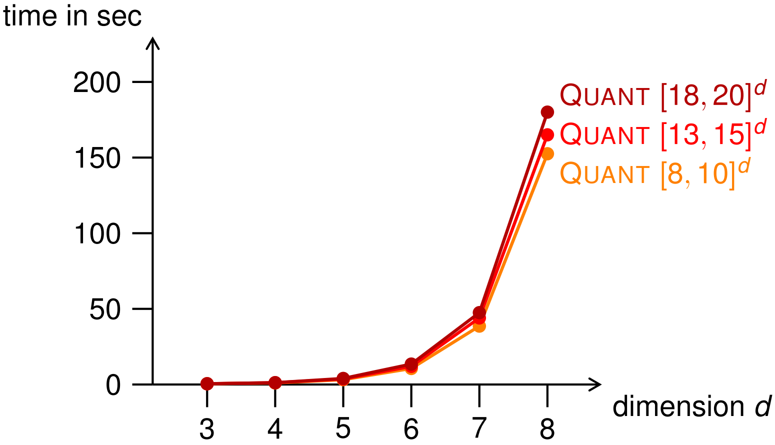




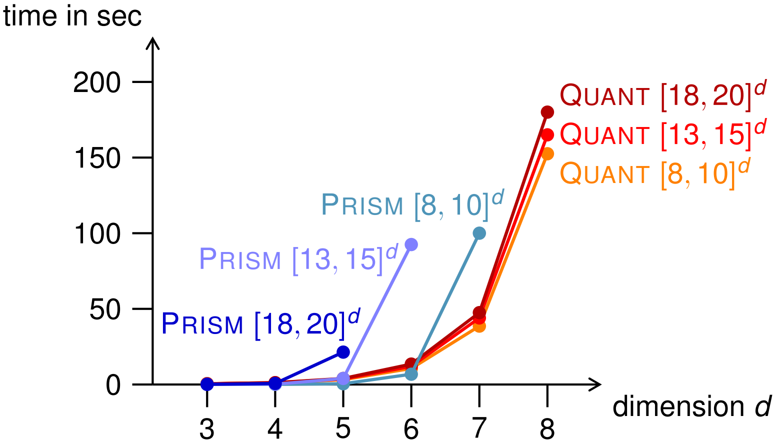
What is Pr(cost $\in$ $[4, 6]^2$)?



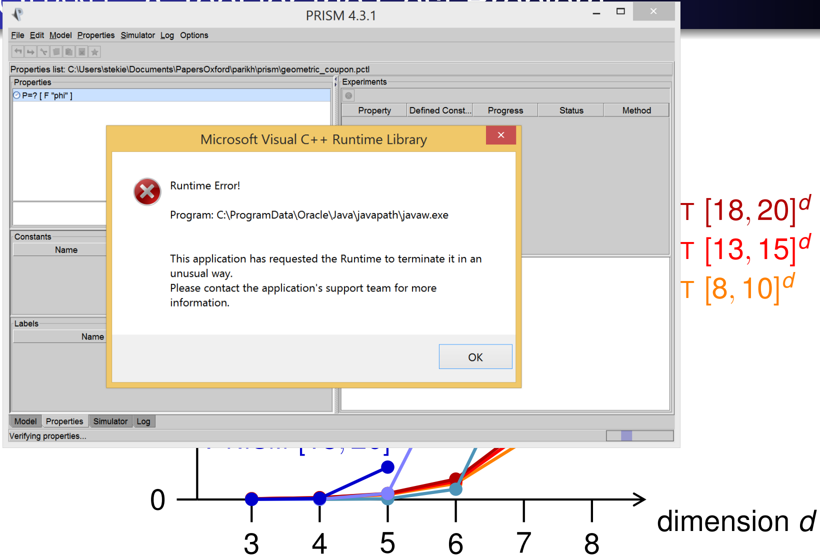
QUANT: A Tool for the Cost Problem



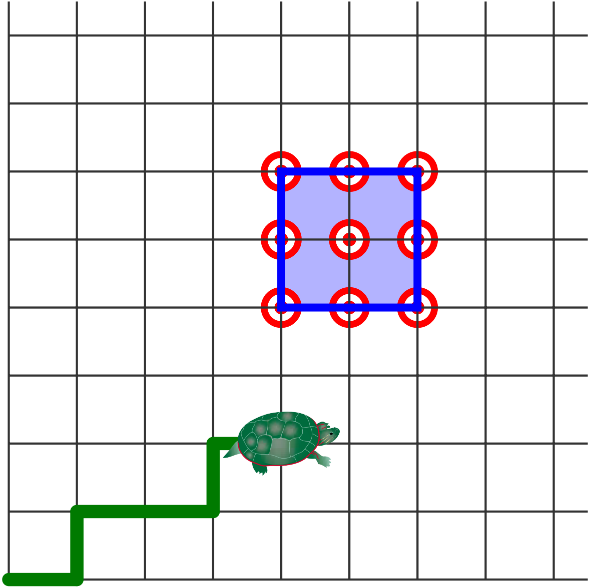
QUANT: A Tool for the Cost Problem



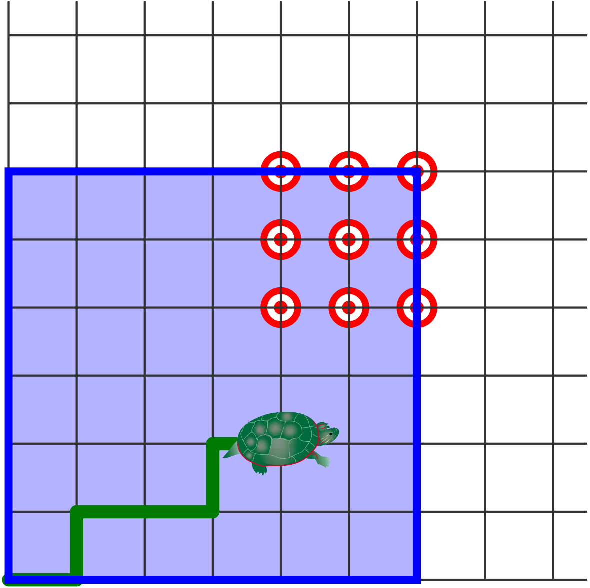
Quant: A Tool for the Cost Problem



QUANT: A Tool for the Cost Problem



QUANT: A Tool for the Cost Problem



Counting Parikh Images

Σ: finite alphabet

$\boldsymbol{p} \in \mathbb{N}^{\Sigma}$: vector

$\mathcal{A}$: language generator (DFA, NFA, CFG)

$N(\mathcal{A}, \boldsymbol{p})$: number of words accepted by $\mathcal{A}$ with Parikh image $\boldsymbol{p}$

Example: for $\mathcal{A} = a^* b a^*$ and $\boldsymbol{p} = (2, 1)$: $N(\mathcal{A}, \boldsymbol{p}) = 3$

PosParikh :=

Input:	Language generators $\mathcal{A}, \mathcal{B}$
	vector $\boldsymbol{p} \in \mathbb{N}^{\Sigma}$
Output:	Is $N(\mathcal{A}, \boldsymbol{p}) > N(\mathcal{B}, \boldsymbol{p})$?

Different variants:

- language generator: DFA, NFA, CFG
- unary or binary encoding of $\boldsymbol{p}$
- fixed or variable alphabet Σ

Results of This Paper: Complexity of PosParikh

vector ***p***	size of Σ	DFA	NFA	CFG
unary encoding	unary	in L	NL-c.	P-c.
	fixed	PL-c.		
	variable	PP-c.		
binary encoding	unary	in L	NL-c.	DP-c.
	fixed	PosMatPow-hard, in CH	PSPACE-c.	PEXP-c.
	variable	PosSLP-hard, in CH		

Results of This Paper: Complexity of PosParikh

<table>
<tr><th>vector p</th><th>size of Σ</th><th>DFA</th><th>NFA</th><th>CFG</th></tr>
<tr><td rowspan="3">unary encoding</td><td>unary</td><td>in L</td><td>NL-c.</td><td>P-c.</td></tr>
<tr><td>fixed</td><td>PL-c.</td><td colspan="2" rowspan="2">PP-c.</td></tr>
<tr><td>variable</td><td></td></tr>
<tr><td rowspan="3">binary encoding</td><td>unary</td><td>in L</td><td>NL-c.</td><td>DP-c.</td></tr>
<tr><td>fixed</td><td>PosMatPow-hard, in CH</td><td rowspan="2">PSPACE-c.</td><td rowspan="2">PEXP-c.</td></tr>
<tr><td>variable</td><td>PosSLP-hard, in CH</td></tr>
</table>

BitParikh $:=$

Input: Language generator $\mathcal{A}$
vector $\boldsymbol{p} \in \mathbb{N}^{\Sigma}$
number $i \in \mathbb{N}$ in binary

Output: Is the i-th bit of $N(\mathcal{A}, \boldsymbol{p})$ equal to 1 ?

PosMatPow

PosMatPow :=

Input: $m \times m$ integer matrix M (in binary)
number $n \in \mathbb{N}$ (in binary)

Output: Is $(M^n)_{1,m} \geq 0$?

The entries of M^n are of exponential size.

Theorem (Galby, Ouaknine, Worrell, 2015)

PosMatPow is in P for $m = 2$.
PosMatPow is in P for $m = 3$ and M given in unary.

PosMatPow reduces to PosSLP.

Theorem (HKL, 2017)

PosMatPow reduces to
*PosParikh with DFAs, binary encoding of **p**, even for $|\Sigma| = 2$.*

From Matrix Powers to Graphs

G: multi-graph with edges labelled by multiplicities

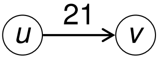
$N(G, u, v, n)$: number of paths from u to v of length n

Lemma

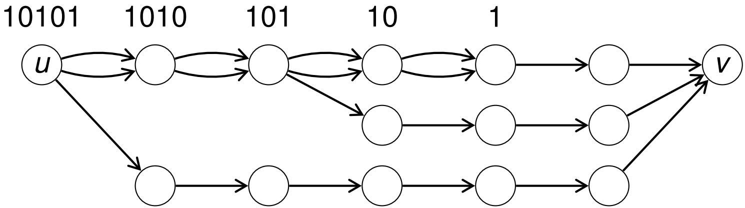
Given an integer matrix M and indices i, j, one can compute in logspace a multi-graph G with vertices u, v^+, v^- such that

$$(M^n)_{i,j} = N(G, u, v^+, n) - N(G, u, v^-, n) \qquad \text{for all } n \in \mathbb{N}.$$

Getting Rid of Edge Weights

Replace 

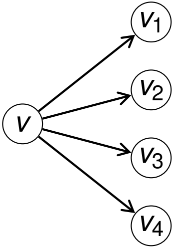
by



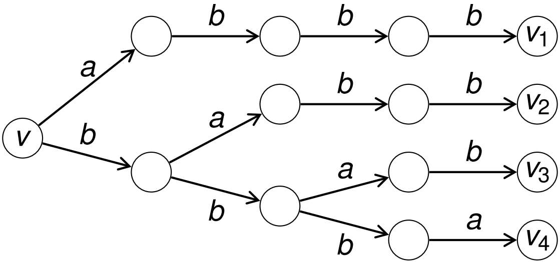
The edge weights are removed at the expense of longer paths.

From Graphs to DFAs

Replace



by



Summary

Counting Parikh images is closely related to performance analysis of probabilistic systems.

The complexity depends strongly on various parameters:

- language generator: DFA, NFA, CFG
- unary or binary encoding of $\boldsymbol{p}$
- fixed or variable alphabet Σ

Results of This Paper: Complexity of PosParikh

<table>
<tr><th>vector p</th><th>size of Σ</th><th>DFA</th><th>NFA</th><th>CFG</th></tr>
<tr><td rowspan="3">unary encoding</td><td>unary</td><td>in L</td><td>NL-c.</td><td>P-c.</td></tr>
<tr><td>fixed</td><td>PL-c.</td><td></td><td></td></tr>
<tr><td>variable</td><td colspan="3">PP-c.</td></tr>
<tr><td rowspan="3">binary encoding</td><td>unary</td><td>in L</td><td>NL-c.</td><td>DP-c.</td></tr>
<tr><td>fixed</td><td>PosMatPow-hard, in CH</td><td rowspan="2">PSPACE-c.</td><td rowspan="2">PEXP-c.</td></tr>
<tr><td>variable</td><td>PosSLP-hard, in CH</td></tr>
</table>