EFFICIENT ALGORITHMS FOR FINDING DIFFERENCES
BETWEEN PROCESS MODELS

A.V. SKOBTSOV, A.A.KALENKOVA

2019 IVANNIKOV ISP RAS OPEN CONFERENCE
DECEMBER 05, 2019

Process mining

The number of information systems around us is constantly growing...

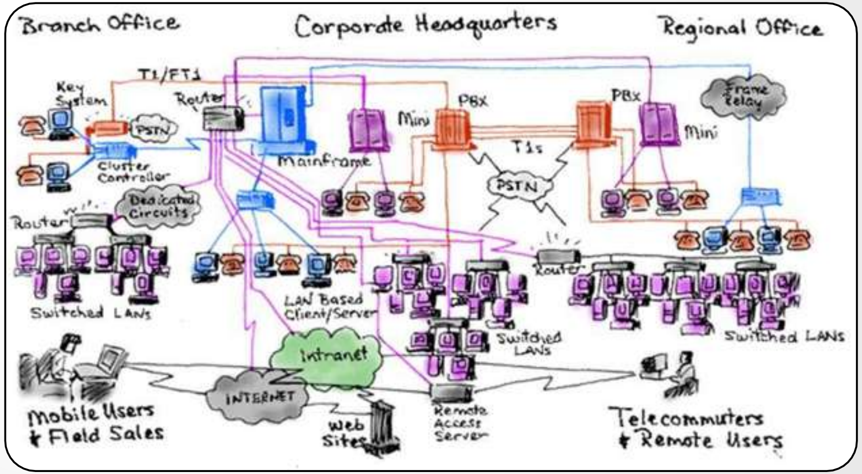
We don't always know how they are used:

- What actions are most often performed?
- In what order?
- Are there bottlenecks?
- How far we are from the expected behavior of the system.

Process mining

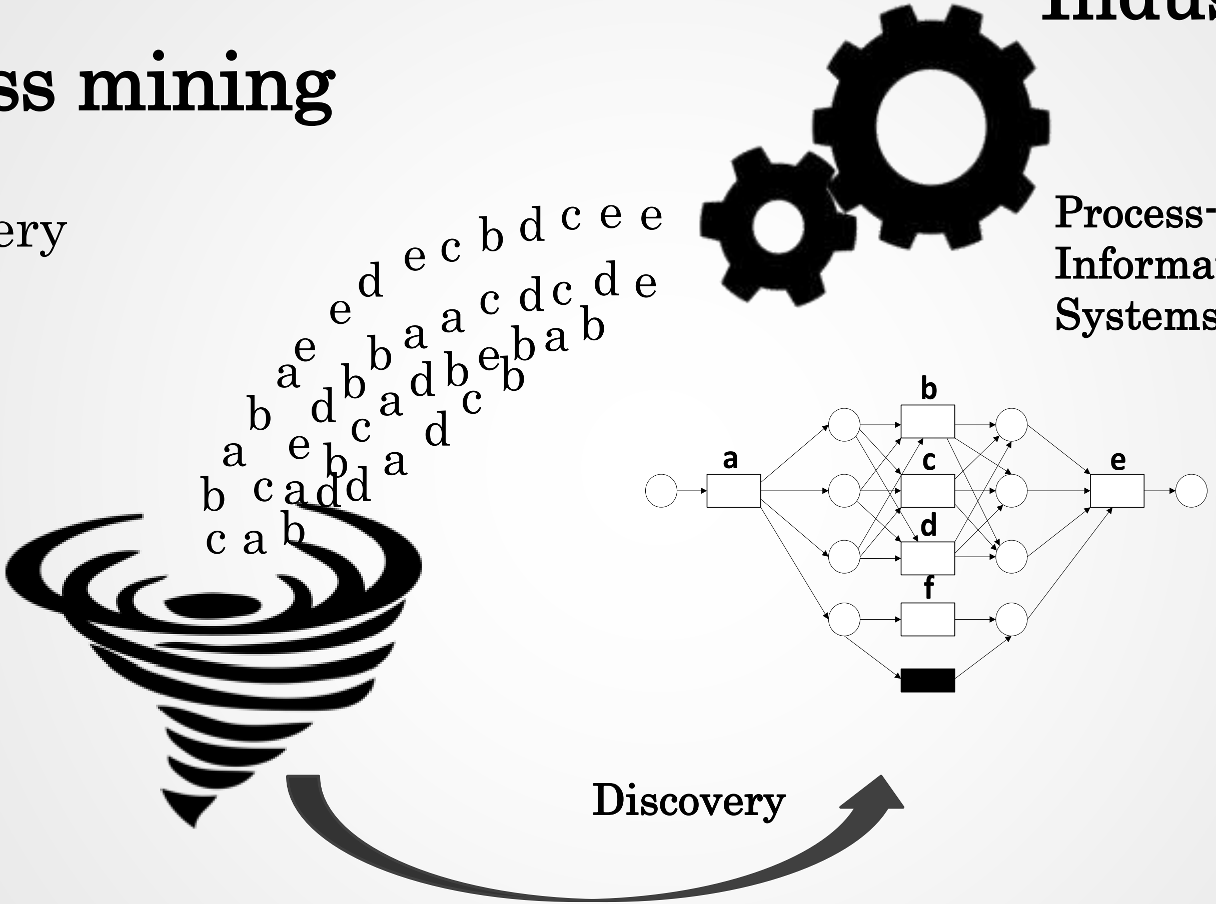
User interaction:
- Internet marketing;
- E-government services;

Complex multi-component systems (software process mining)



Process mining

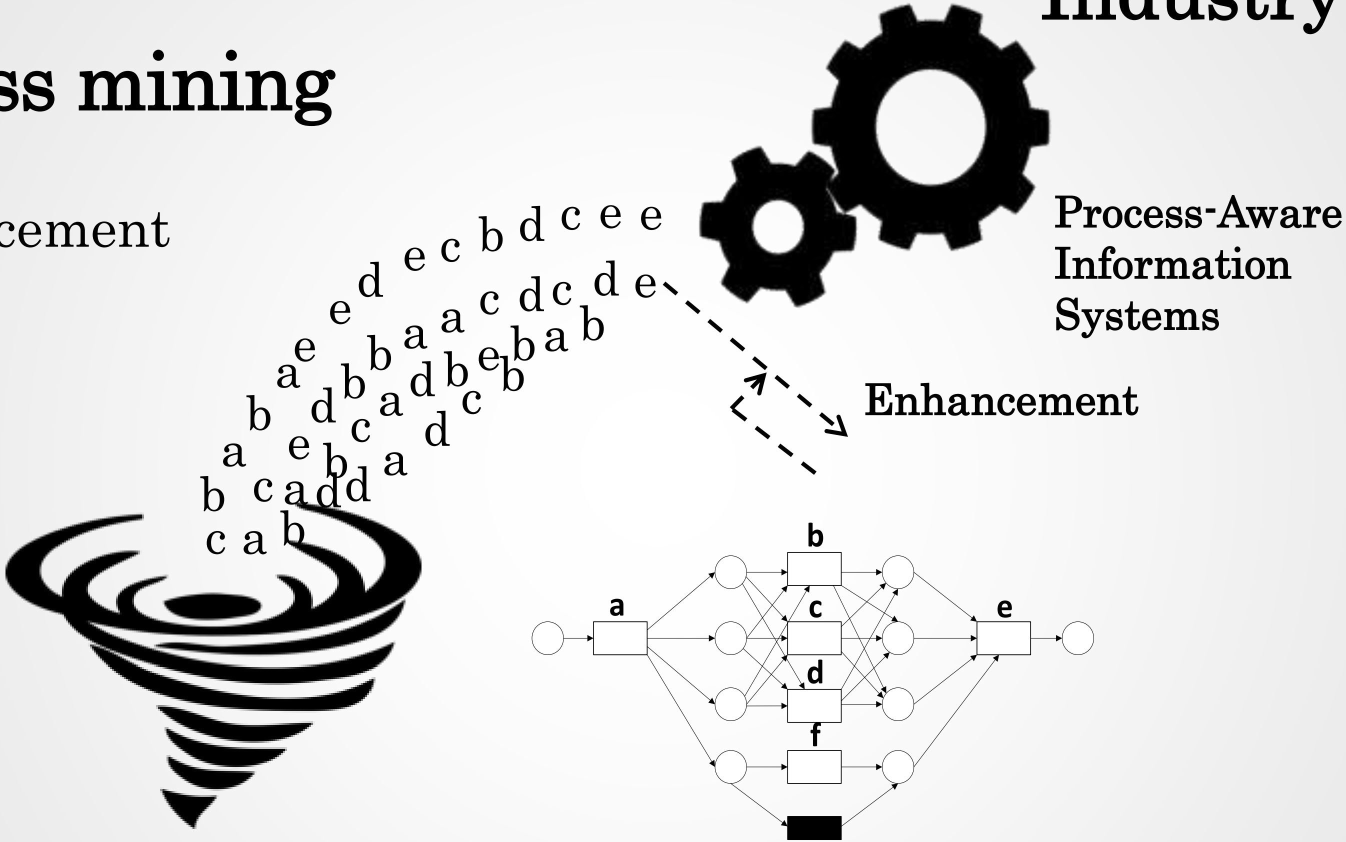
- Discovery



Industry

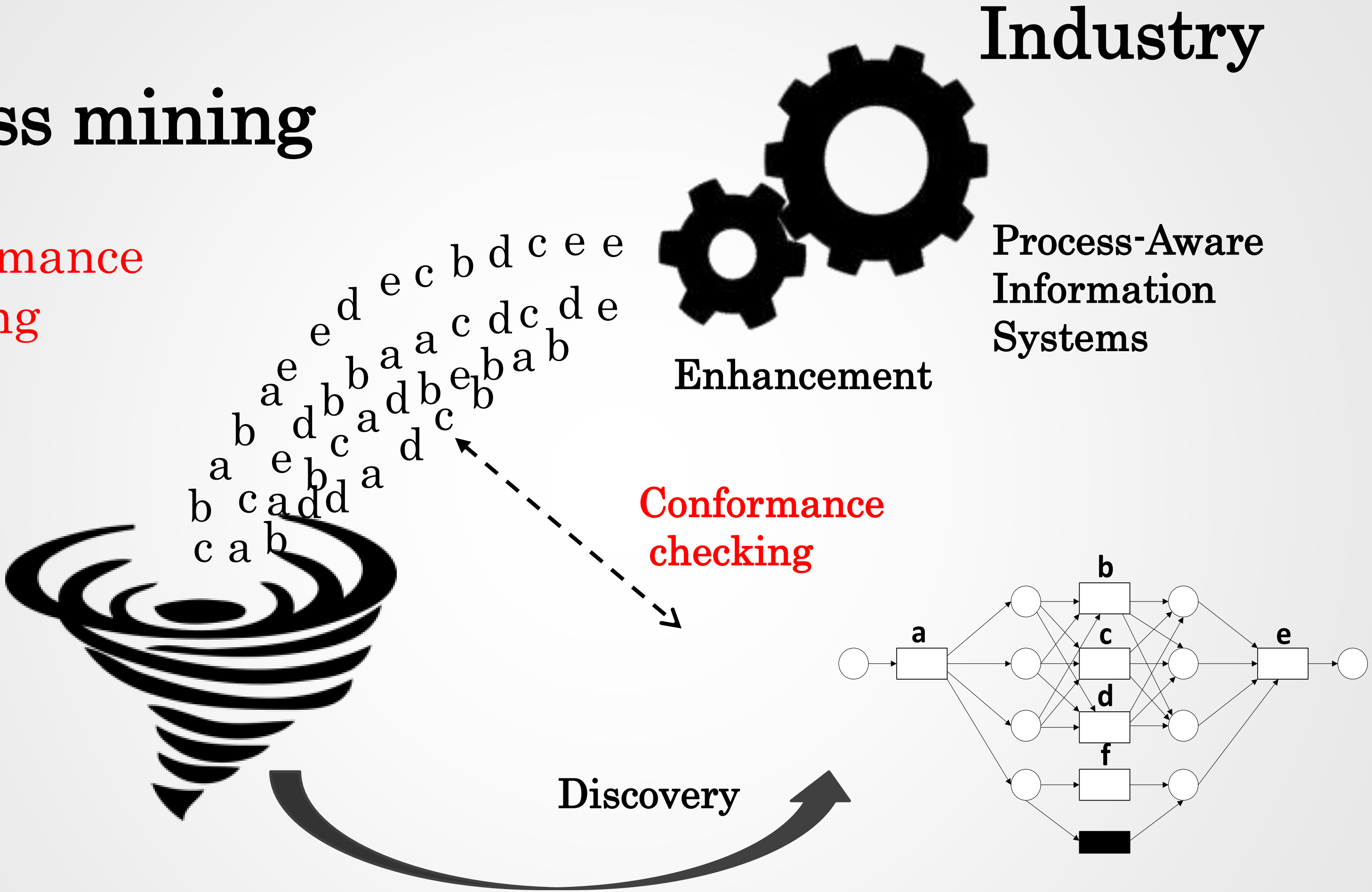
Process mining

- Enhancement



Process mining

- Conformance checking



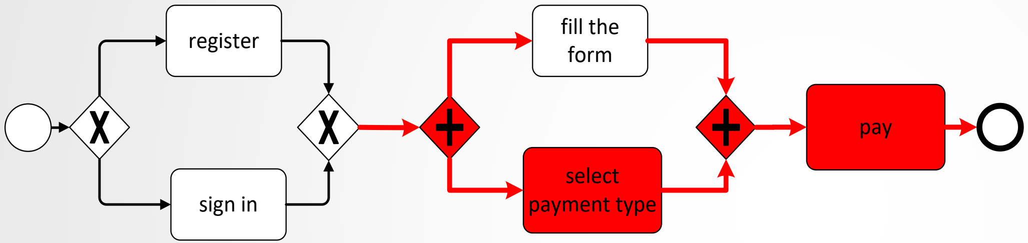
Process mining. Conformance checking

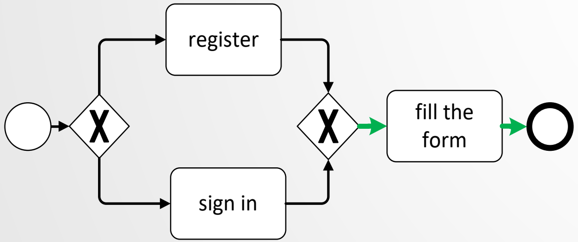
L2L – Comparison of event structures

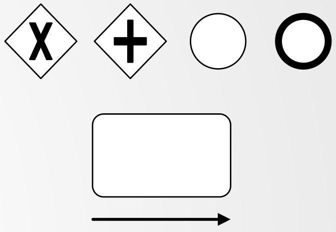
L2M – Replay techniques

M2M - Must be something visual?

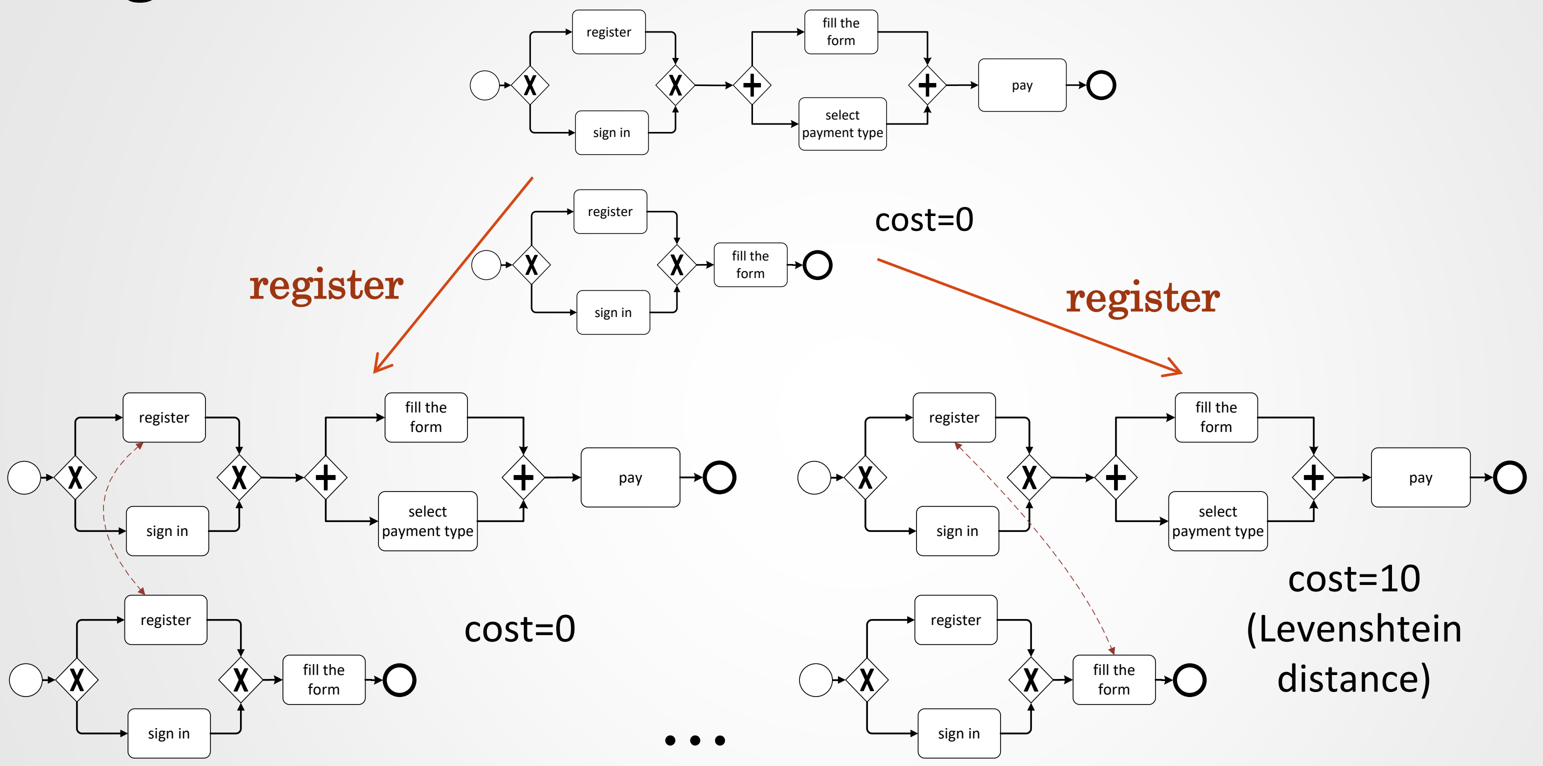
Finding Minimal Graph Edit Distance



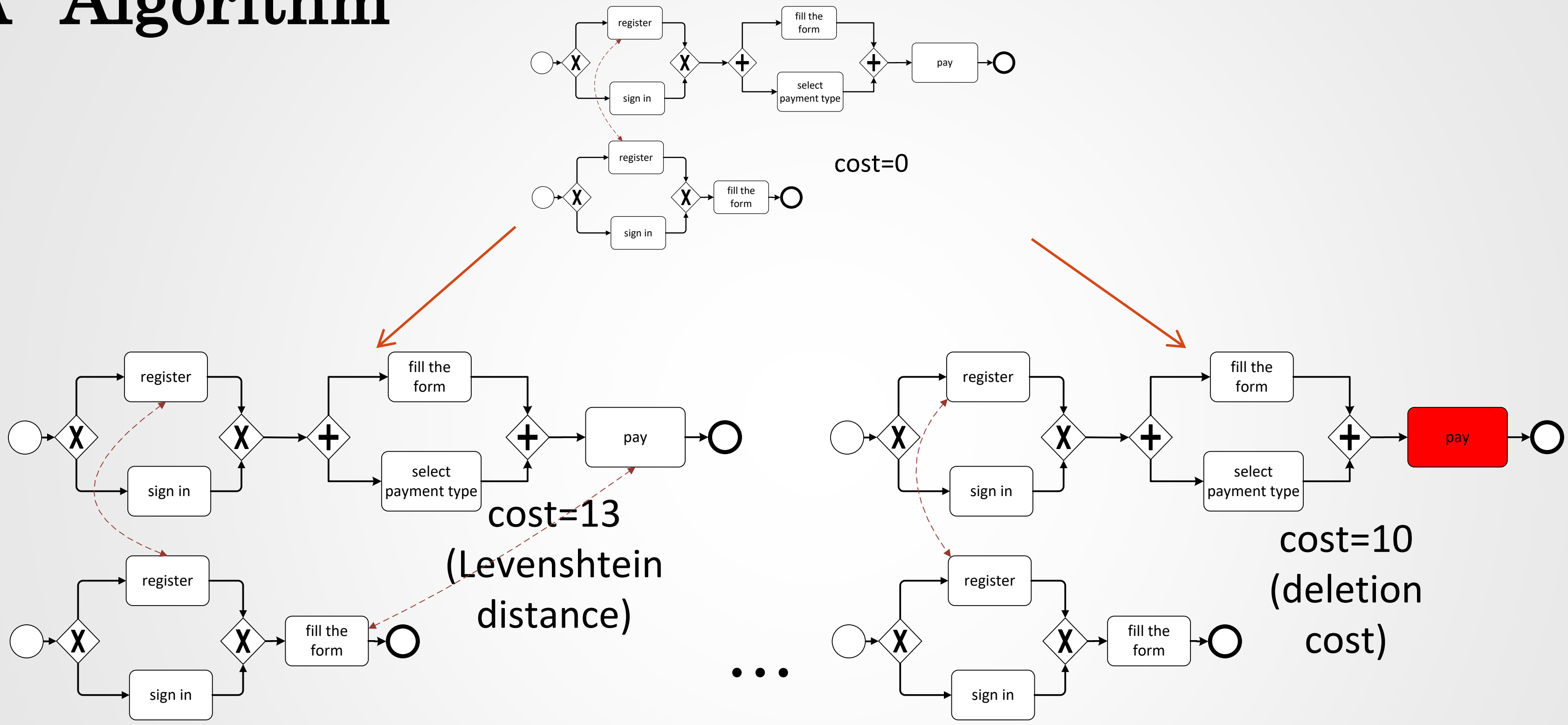




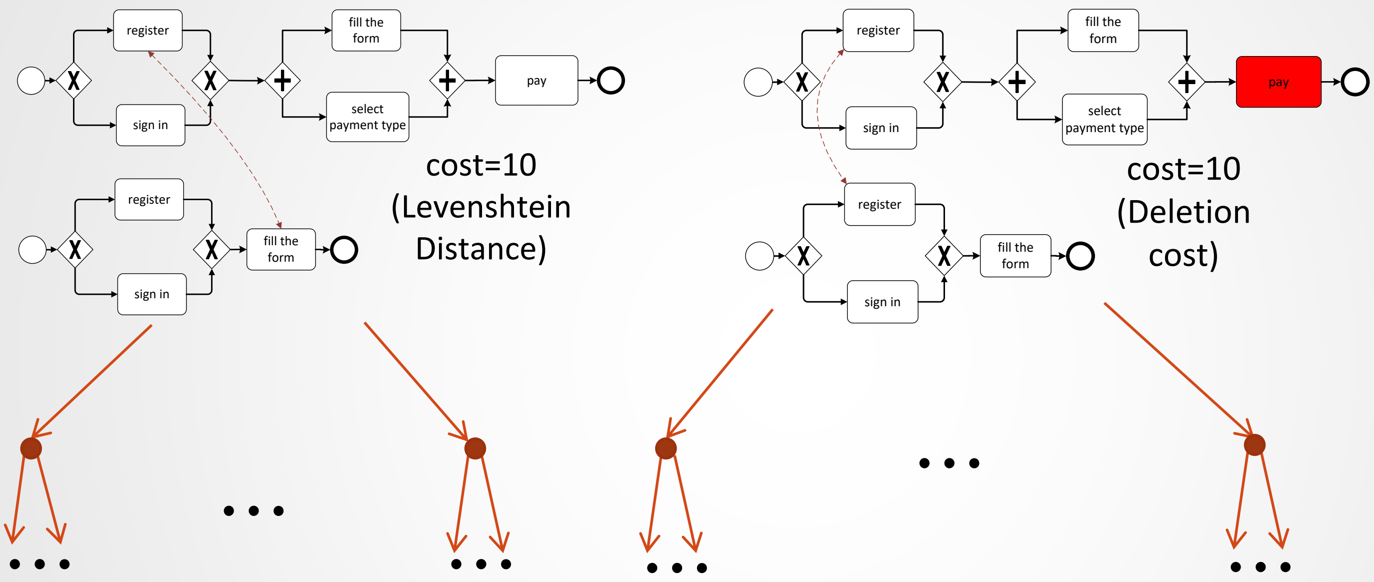
A* Algorithm


register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=0
register
register
register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=0
...
register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=10
(Levenshtein
distance)

A* Algorithm

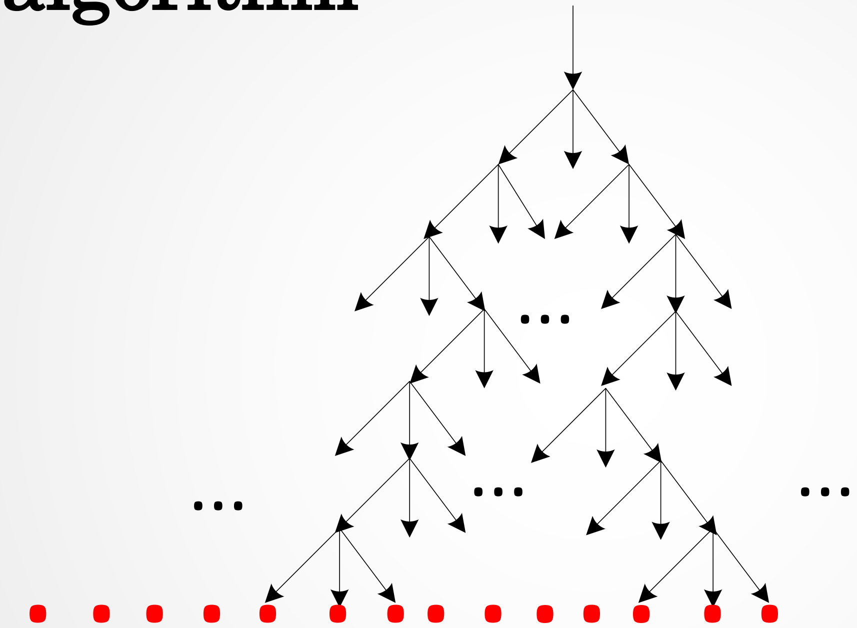

register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=0
register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=13 (Levenshtein distance)
. . .
register
sign in
fill the form
select payment type
pay
register
sign in
fill the form
cost=10 (deletion cost)

A* Algorithm



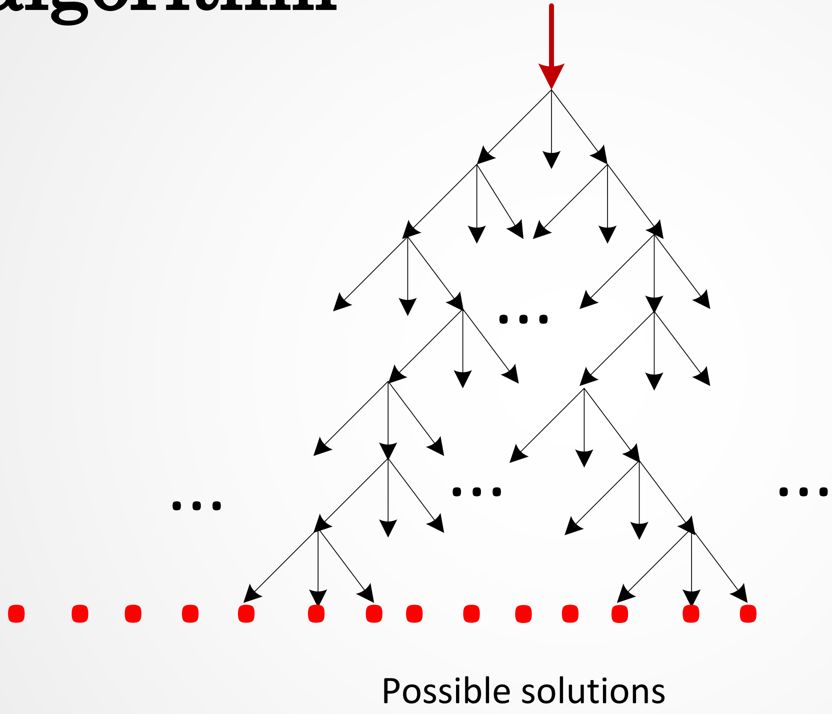
NP problem

Greedy algorithm

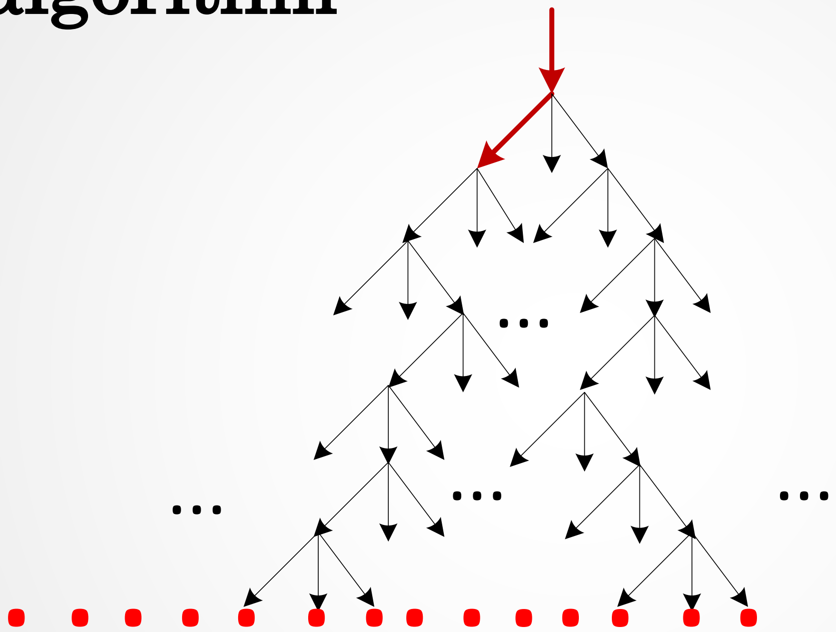


Possible solutions

Greedy algorithm

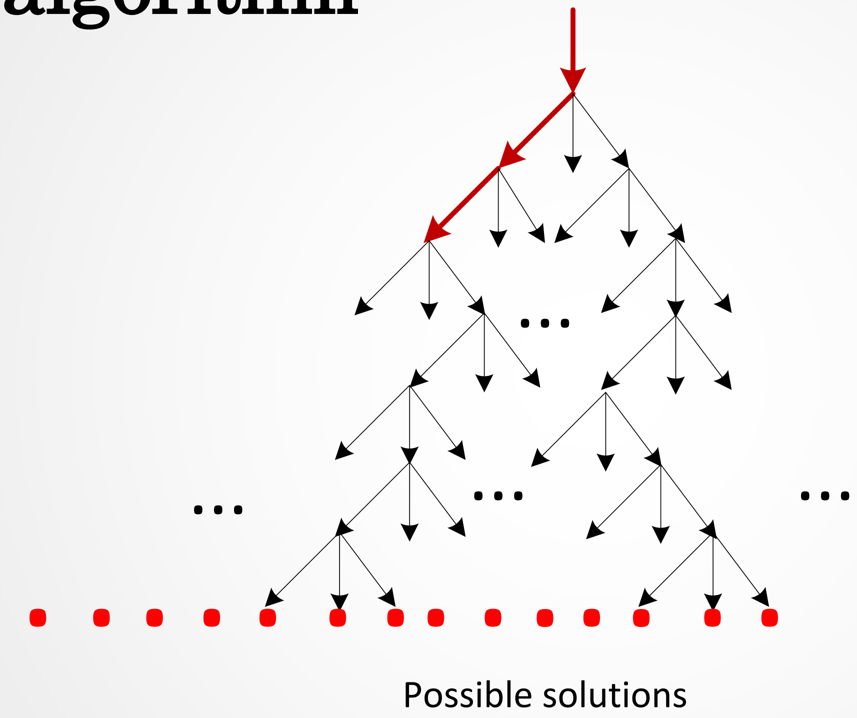


Greedy algorithm

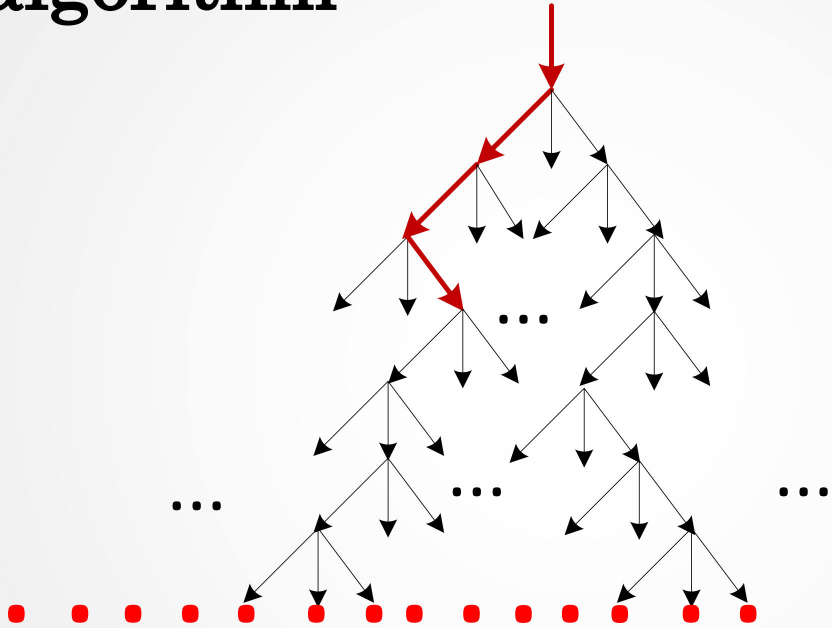


Possible solutions

Greedy algorithm

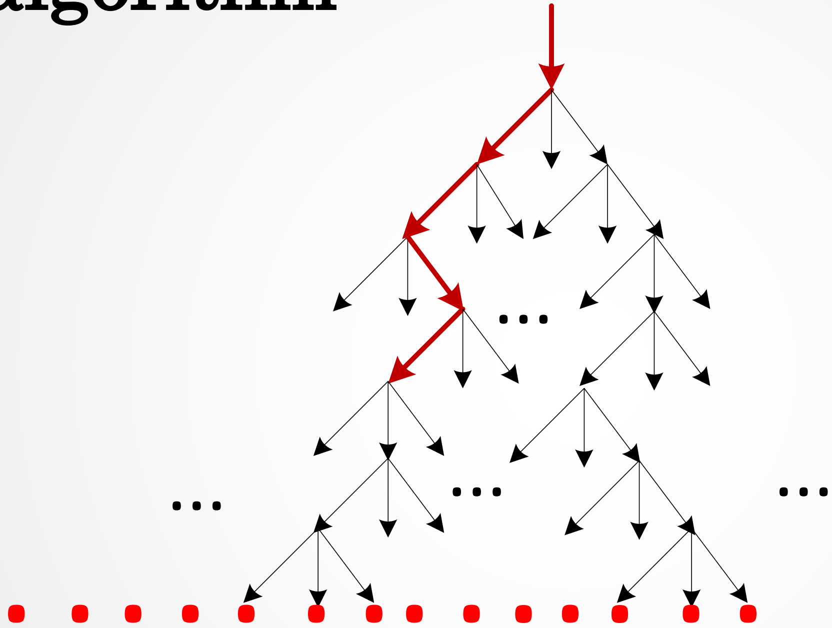


Greedy algorithm



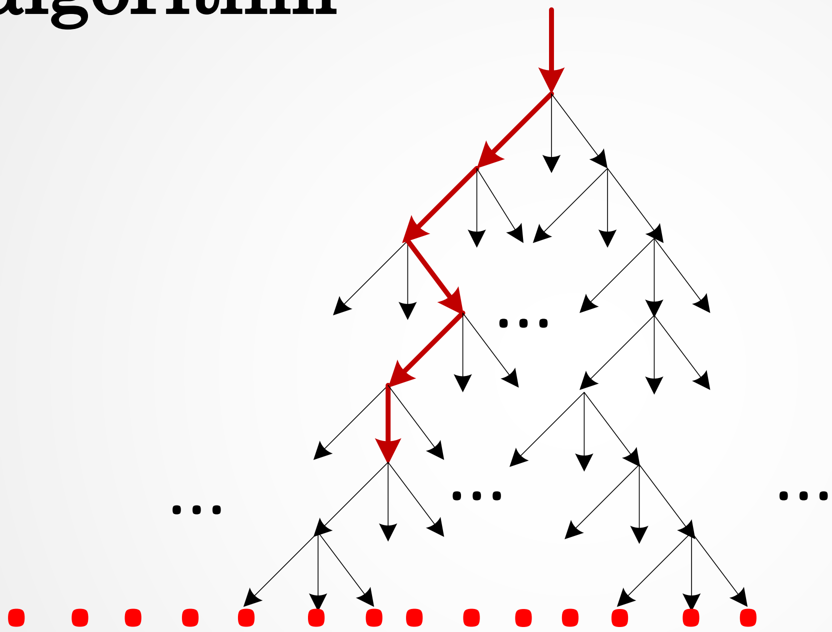
Possible solutions

Greedy algorithm



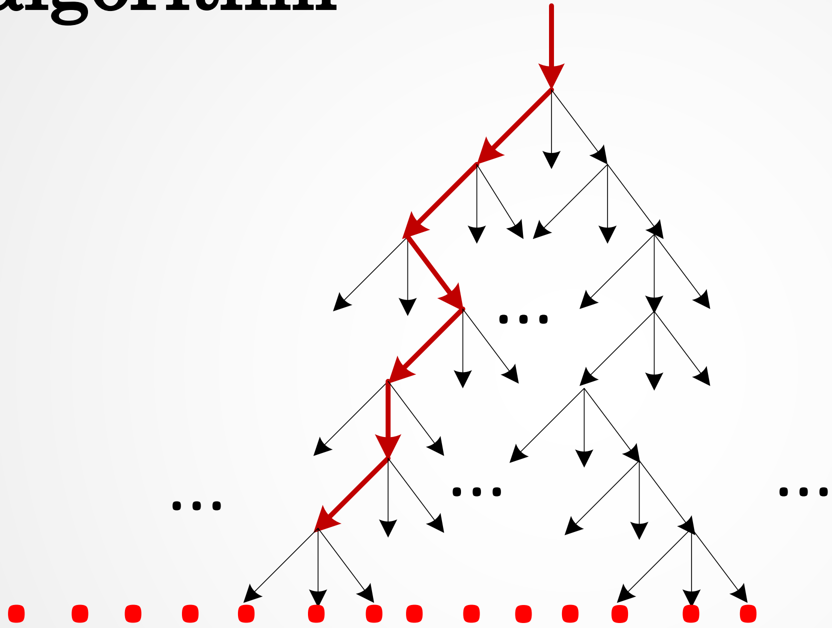
Possible solutions

Greedy algorithm



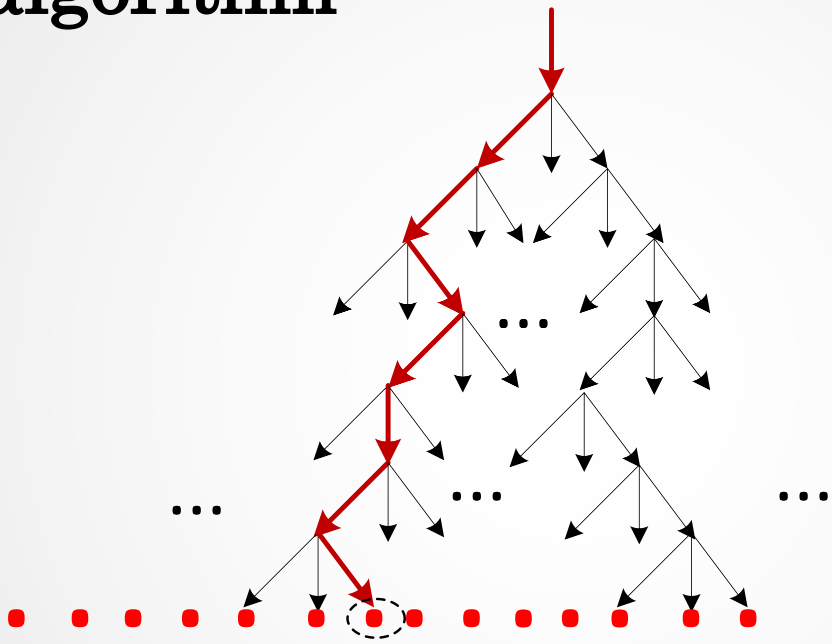
Possible solutions

Greedy algorithm



Possible solutions

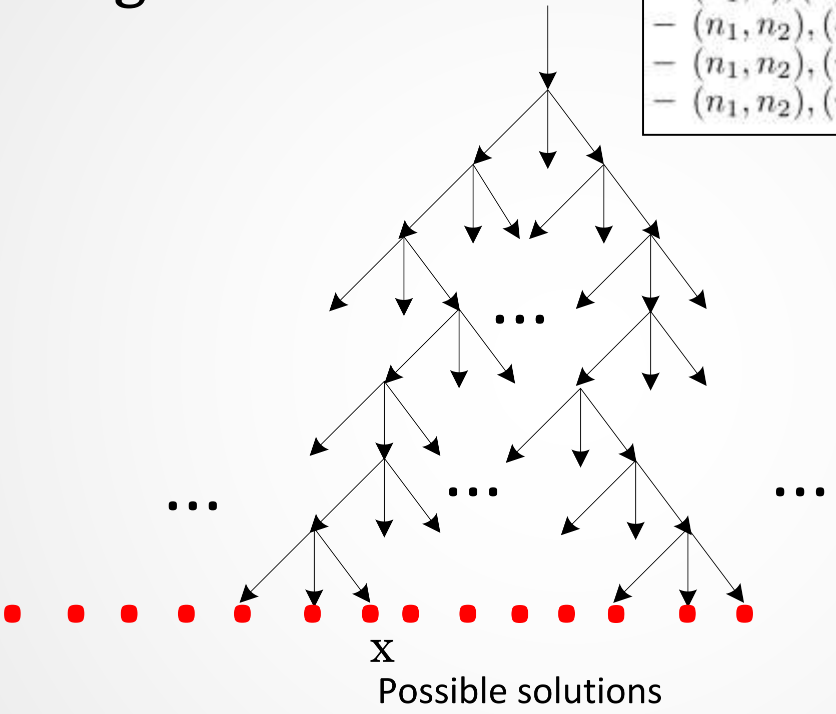
Greedy algorithm



Possible solutions

Tabu search algorithm

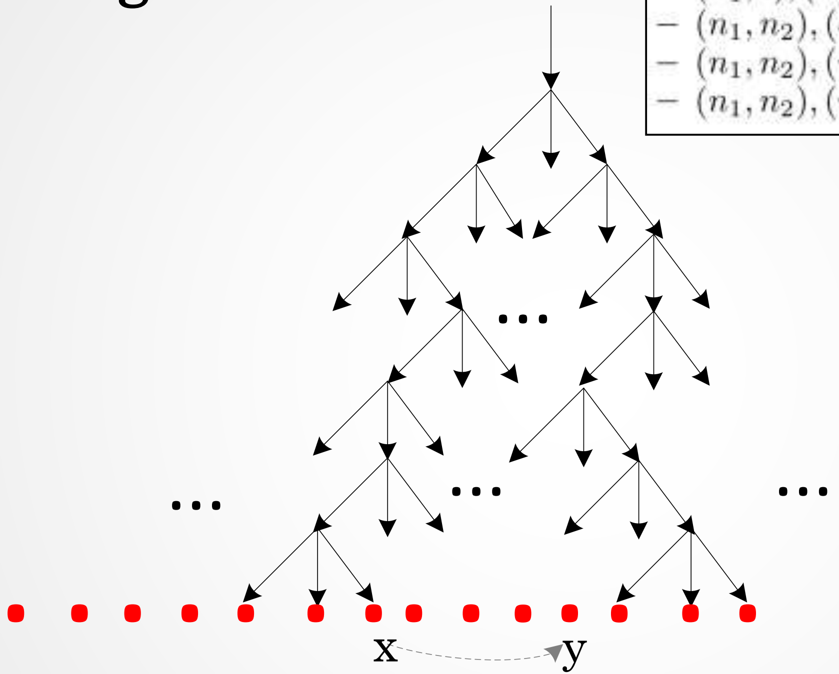
- $(n_1, n_2) \rightarrow (n_1, \epsilon), (\epsilon, n_2)$,
- $(n_1, \epsilon), (\epsilon, n_2) \rightarrow (n_1, n_2)$,
- $(n_1, \epsilon), (n'_1, n_2) \rightarrow (n_1, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (\epsilon, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, \epsilon), (n'_1, n_2), (\epsilon, n'_2)$.



Tabu list =<x>

Tabu search algorithm

- $(n_1, n_2) \rightarrow (n_1, \epsilon), (\epsilon, n_2)$,
- $(n_1, \epsilon), (\epsilon, n_2) \rightarrow (n_1, n_2)$,
- $(n_1, \epsilon), (n'_1, n_2) \rightarrow (n_1, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (\epsilon, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, \epsilon), (n'_1, n_2), (\epsilon, n'_2)$.

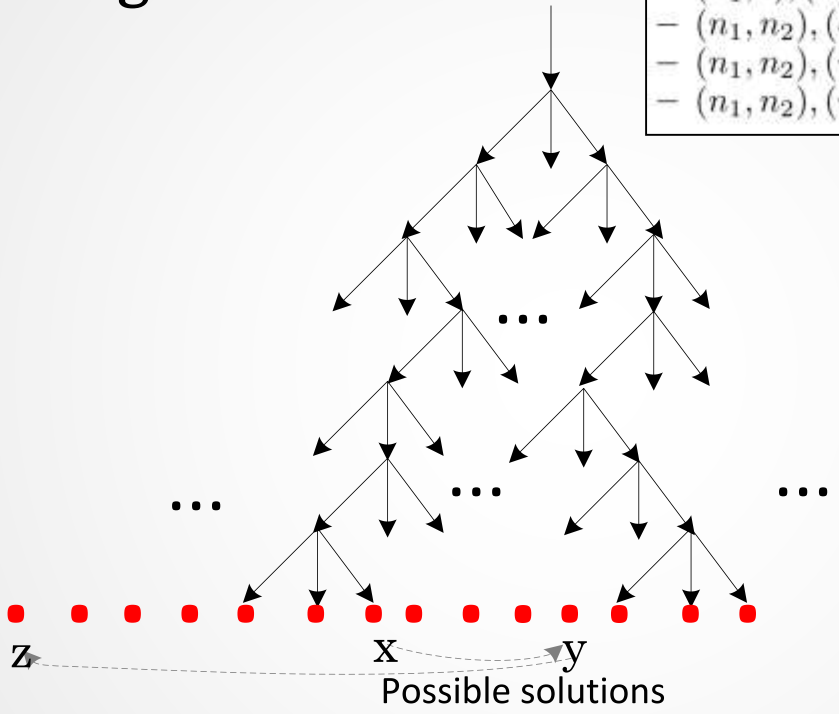


Possible solutions

Tabu list =<x, y>

Tabu search algorithm

- $(n_1, n_2) \rightarrow (n_1, \epsilon), (\epsilon, n_2)$,
- $(n_1, \epsilon), (\epsilon, n_2) \rightarrow (n_1, n_2)$,
- $(n_1, \epsilon), (n'_1, n_2) \rightarrow (n_1, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (\epsilon, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, n'_2), (\epsilon, n_2), (n'_1, \epsilon)$,
- $(n_1, n_2), (n'_1, n'_2) \rightarrow (n_1, \epsilon), (n'_1, n_2), (\epsilon, n'_2)$.



Tabu list =<x, y, z>

Simulated Annealing algorithm

Data: $G_1 = (N_1, E_1, t_1, l_1)$ and $G_2 = (N_2, E_2, t_2, l_2)$ – business process graphs; $maxTemperature$ – maximal temperature; $temperatureDec$ – temperature decreasing step;

Result: graph edit distance between G_1 and G_2;

initialize R_{cur} – *edit relation*;

$R_{cur} \leftarrow R_{greedy}$;

$T_{cur} \leftarrow maxTemperature$;

while $(T_{cur} > 0)$ **do**

  $generateOneStepVariants(R_{cur})$;

  $variant \leftarrow takeRandom(oneStepVariants)$;

  if $P(R_{cur}, variant, T_{cur}) \geq random(0, 1)$ **then**

    $R_{cur} \leftarrow variant$;

  end

  $T_{cur} \leftarrow T_{cur} - temperatureDec$;

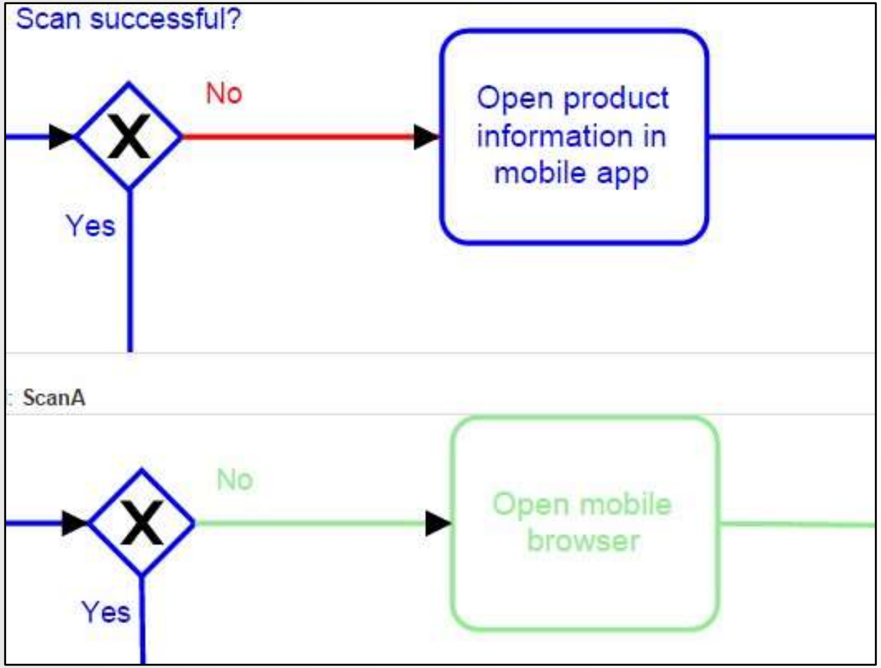
end

return $cost(R_{cur})$;

Ant Colony algorithm

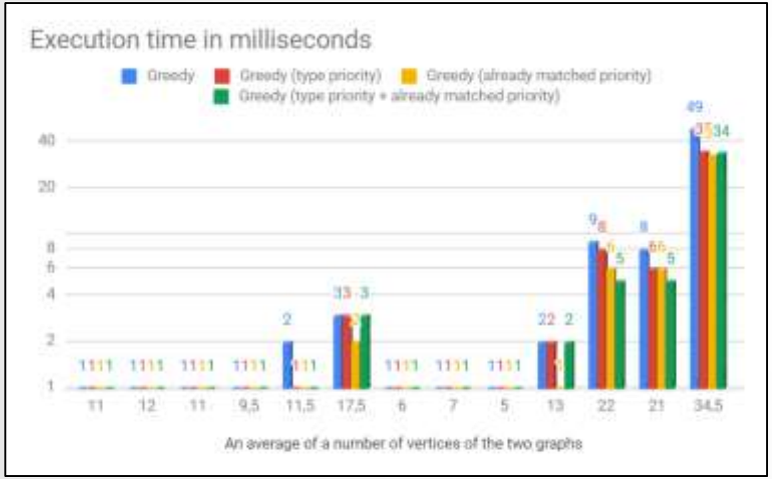
1. Initialize pheromone map with the initial value
2. Generate N ants; each ant does the following:
    1. Generates all the possible vertex replacements for the current state
    2. Calculates the cost for each vertex replacement by the formulae:
    $$\frac{edgePheromones^{pheromonePower}}{pathCost^{distancePower}}$$
    where $edgePheromones$ – the cost of replacing vertices (taken from the pheromone map);
    $pathCost$ – sum of all the edges and vertex replacements
    3. Selects a random replacement with the probability of each replacement:
    $$\frac{replacementCost}{sumOfAllCosts}$$
    4. Performs steps 1-3 until all the vertices are processed
3. Change all the pheromone values by formulae: $(1 - pheromoneEvaporation) * currentValue$
4. For all solutions from the step 2 do the following:
    1. Increase the pheromone map's value for the replacement by $\frac{distanceCoeff}{replacementCost}$
5. Repeat steps 2-3 predefined number of times

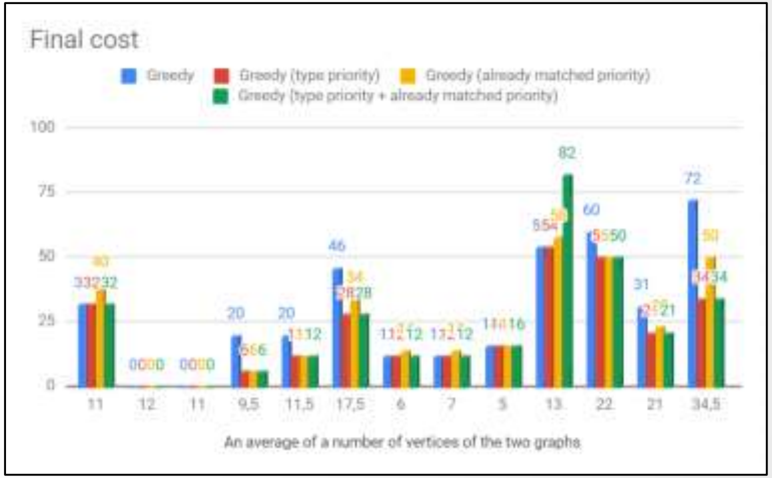
BPMNDiffViz Tool



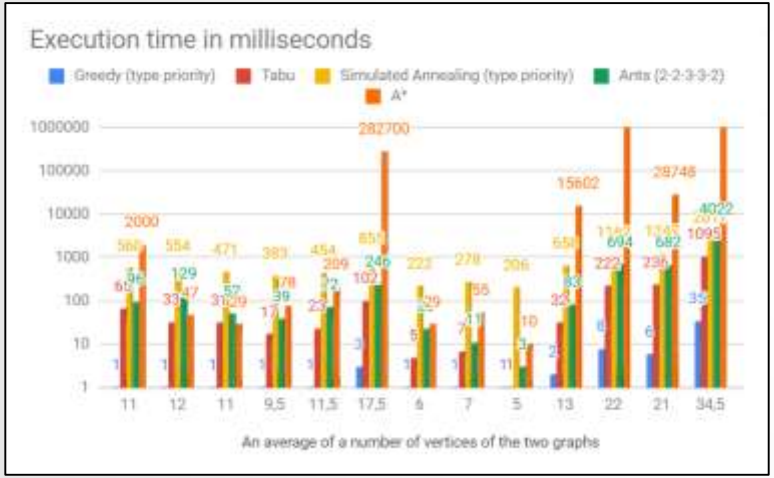
https://pais.hse.ru/research/projects/CompBPMN

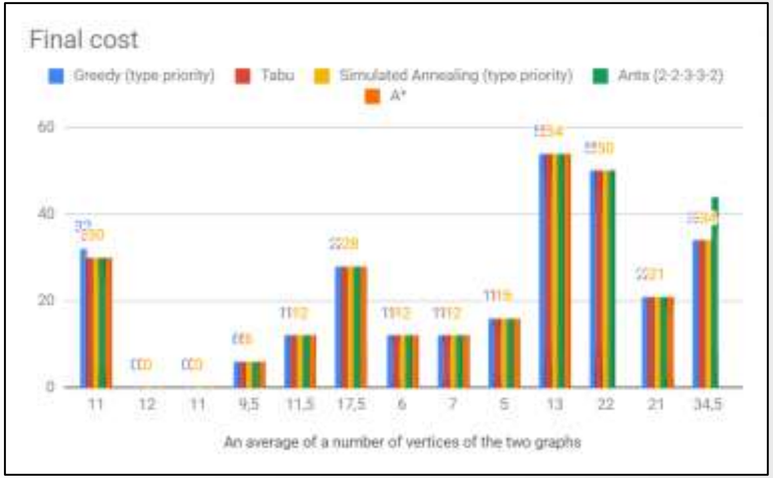
Experimental results. BPMN models discovered from artificial event logs (different algorithms)



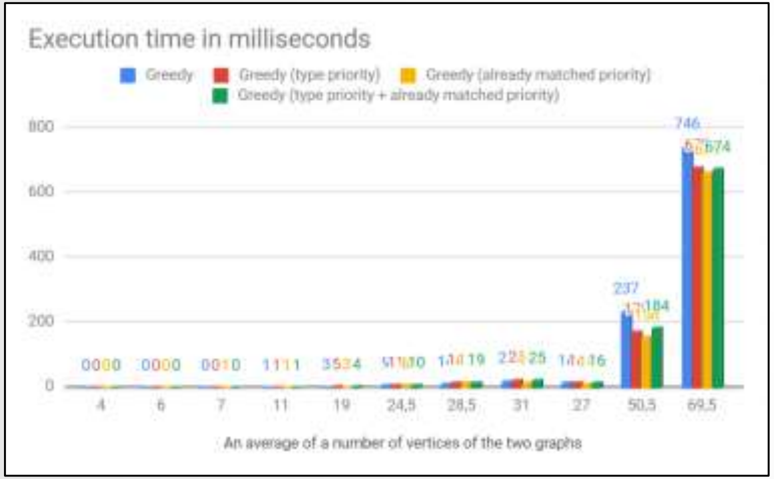


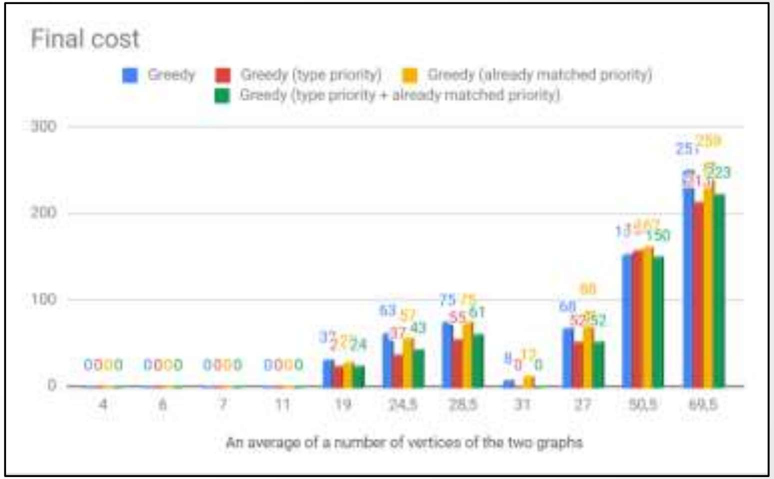
Experimental results. BPMN models discovered from artificial event logs (different algorithms)



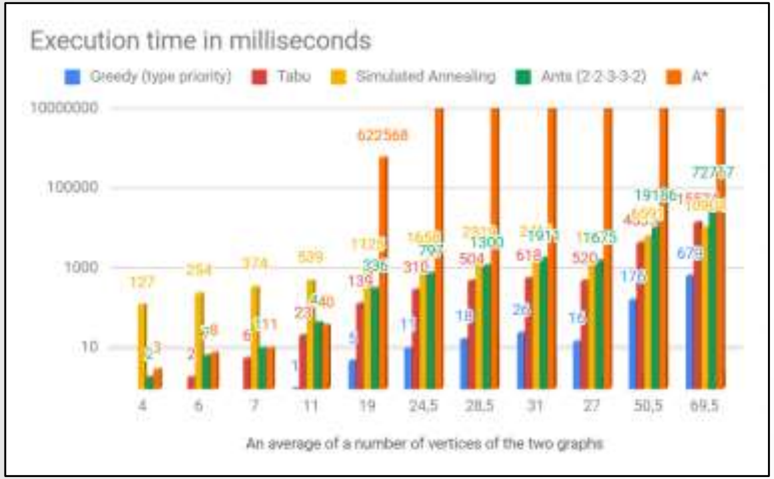


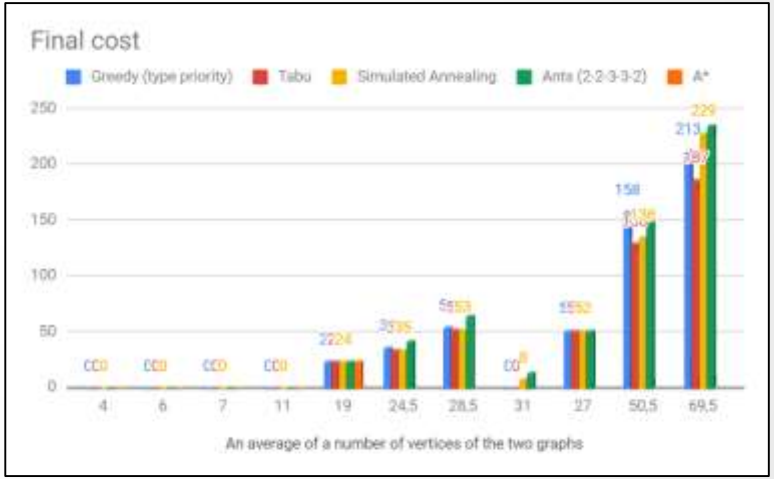
Experimental results. BPMN models discovered from real event logs (different parts of logs)





Experimental results. BPMN models discovered from real event logs (different parts of logs)





Future work

Industry

- ✓ New suboptimal methods
- ✓ New application fields

Theory

- ✓ Compare with other conformance checking methods
- ✓ Different discovery algorithms (different structure of process models)

Thank you!

A.V. Skobtsov: andrewskobcov@yandex.ru
A.A. Kalenkova: akalenkova@hse.ru