HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Branching

Teppo Niinimäki

‹teppo.niinimaki@helsinki.fi›

October 11, 2011

University of Helsinki

Department of Computer Science

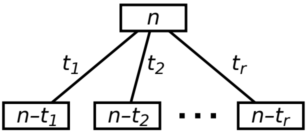
Outline

1. Branching algorithms in general
2. ~~The *k*-Satisfiability problem~~
3. The Maximum Independent Set problem

Introduction

Given a problem of size *n*.
Two types of (polynomial time) rules:

- *reduction rules*
  - simplify the problem or
  - halt

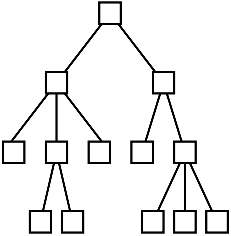


- *branching rules*
  - recursively smaller instances

Search tree

Search tree

- models the execution of algorithm
- exponential number of nodes

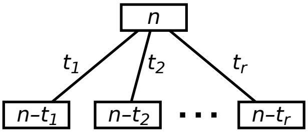


Running time

- polynomial factors ignored
- $\mathcal{O}^*(\text{number of nodes}) = \mathcal{O}^*(\text{number of leaves})$

Branching rules

For a branching rule b



- branching vector $\mathbf{b} = (t_1, t_2, \ldots, t_r)$
- for max. number of leaves $T(n)$ holds:

$$T(n) \leq T(n - t_1) + T(n - t_2) + \ldots + T(n - t_r)$$

  solution: $T(n) = \alpha^n$ for some $\alpha > 1$

- branching factor $\tau(t_1, t_2, \ldots, t_r) = \alpha$

$\Rightarrow$ running time $\mathcal{O}^*(\alpha^n)$ if only b used

Branching factors

Common binary branching factors $\tau(i, j)$:

	1	2	3	4	5	6
1	2.0000	1.6181	1.4656	1.3803	1.3248	1.2852
2	1.6181	1.4143	1.3248	1.2721	1.2366	1.2107
3	1.4656	1.3248	1.2560	1.2208	1.1939	1.1740
4	1.3803	1.2721	1.2208	1.1893	1.1674	1.1510
5	1.3248	1.2366	1.1939	1.1674	1.1487	1.1348
6	1.2852	1.2107	1.1740	1.1510	1.1348	1.1225

Example: $\tau(2, 3) \approx 1.3248$

Generally for $\mathbf{b} = (t_1, t_2, \ldots, t_r)$:

- the order of t_i:s irrelevant
- larger elements $t_i \Rightarrow$ smaller branching factor

Multiple branching rules

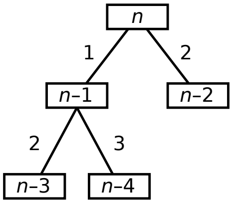
General case:

- branching rules $b_1, b_2, \ldots, b_k$
- branching factors $\alpha_1, \alpha_2, \ldots, \alpha_k$
- running time $\mathcal{O}^*(\alpha^n)$ where $\alpha = \max_i \alpha_i$

Addition of branching vectors:

- Example: after $\mathbf{b} = (1, 2)$
  - left branch: $\mathbf{b}' = (2, 3)$
  - right branch: any rule

$\Rightarrow$ combined branching rule: $(3, 4, 2)$



Outline

1. Branching algorithms in general
2. ~~The *k*-Satisfiability problem~~
3. The Maximum Independent Set problem

Maximum Independent Set

Given an undirected graph $G = (V, E)$.

- $I \subseteq V$ is *independent set* if no vertices of I are adjacent.

The MAXIMUM INDEPENDENT SET problem:
"Find an independent set of maximum size."

The size of the problem: $n = |V|$

Notation

For node $v \in V$

- $N(v)$ neighborhood, $N[v]$ closed neighborhood
- $N^2(v)$ vertices at distance of 2 from v
- $G \setminus v$ subgraph with v removed

For nodes $S \subset V$

- $G[S]$ subgraph induced by S
- $G \setminus S$ subgraph induced by $V \setminus S$

For graph G

- $\delta(G)$ minimum degree of G
- $\Delta(G)$ maximum degree of G

MIS algorithm

Input: A graph $G = (V, E)$.
Output: A maximum independent set of G.

- **if** $V = \emptyset$ **then** return $\emptyset$
- **if** $\delta(G) = 0$ **then** ...
- **if** $\delta(G) = 1$ **then** ...
- **if** $\delta(G) = 2$ **then** ...
- **if** $\delta(G) = 3$ **then** ...
- **if** $\Delta(G) \geq 6$ **then** ...
- **if** G is disconnected **then** ...
- **if** G is 4 or 5-regular **then** ...
- **if** $\Delta(G) = 5$ and $\delta(G) = 4$ **then** ...

Observations

Let $v \in V$ and $\mathrm{mis}(G)$ be some maximum independent set of G.

Lemma (2.6). If no maximum independent set contains v then every maximum independent set contains at least two vertices from $N(v)$.

Simplicial rule: If $N[v]$ is a clique, then $\{v\} \cup \mathrm{mis}(G \setminus N[v])$ is a maximum independent set of G.

The running time

Worst case branching factors:

- $V = \emptyset \quad \Rightarrow \quad$ reduction
- $\delta(G) = 0 \quad \Rightarrow \quad$ reduction
- $\delta(G) = 1 \quad \Rightarrow \quad$ reduction
- $\delta(G) = 2 \quad \Rightarrow \quad \alpha < 1.1939$
- $\delta(G) = 3 \quad \Rightarrow \quad \alpha < 1.2721$
- $\Delta(G) \geq 6 \quad \Rightarrow \quad \alpha < 1.2445$
- G is disconnected $\quad \Rightarrow \quad \alpha < 1.1893$
- G is 4 or 5-regular $\quad \Rightarrow \quad$ ignored
- $\Delta(G) = 5$ and $\delta(G) = 4 \quad \Rightarrow \quad \alpha < 1.2786$

$\Rightarrow$ The running time is $\mathcal{O}^*(1.2786^n)$.

Conclusion

Branching algorithms:

- Based on recursive division to smaller subproblems
- Running time might be much better for some particular instances
- Typically low (polynomial / linear) space complexity