Phase transitions in combinatorial optimization problems
Course at Helsinki Technical University, Finland, autumn 2007
by Alexander K. Hartmann (University of Oldenburg)
Lecture 3, 25. September 2007

Better situation for:

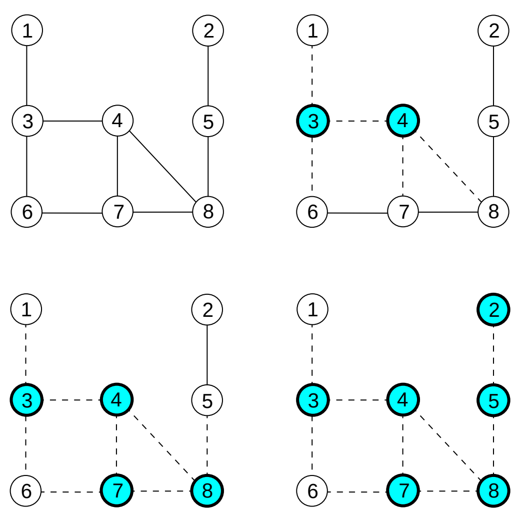
algorithm 2-approximation($G = (V, E)$)
begin
  initialize $V_{\text{vc}} = \emptyset$;
  initialize $M = \emptyset$;
  while there are *uncovered* edges (i. e., $E \neq \emptyset$) **do**
  begin
    take one arbitrary edge $\{i, j\} \in E$;
    mark i and j as *covered*: $V_{\text{vc}} = V_{\text{vc}} \cup \{i, j\}$;
    add $\{i, j\}$ to the matching: $M = M \cup \{\{i, j\}\}$;
    remove from E all edges incident to i or j;
  end;
  return(V_{vc});
end

Example: 2-Approximation heuristic

It. 1 (say) edge $\{3, 4\} \to V_{\text{vc}} = \{3, 4\}$, $M = \{\{3, 4\}\}$
$\{1, 3\}$, $\{3, 4\}$, $\{3, 6\}$, $\{4, 7\}$ and $\{4, 8\}$ are covered

It. 2 $\{7, 8\} \to V_{\text{vc}} = \{3, 4, 7, 8\}$, $M = \{\{3, 4\}, \{7, 8\}\}$
also $\{5, 8\}$, $\{6, 7\}$ and $\{7, 8\}$ are covered

It. 3 Only edge $\{2, 5\}$ is left $\to V_{\text{vc}} = \{2, 3, 4, 5, 7, 8\}$, $M = \{\{3, 4\}, \{7, 8\}, \{2, 5\}\}$.



Note 1: For order $\{1,3\}$, $\{2,5\}$, $\{6,7\}$ and $\{4,8\}$ $V_{vc} = V$ twice the size of minimum VC.

Note 2: never be able to "find" the minimum VC: e. g., $V_{\text{vc}}^{\text{min}} = \{3,5,7,8\}$. □

Theorem: size $|V_{\text{vc}}| \leq 2|V_{\text{vc}}^{\text{min}}|$.
Proof:
Algorithm also constructs matching M. Since two vertices in V_{vc} for each edge in M $\rightarrow$

$$|V_{\text{vc}}| = 2|M|\,. \tag{1}$$

Since (by Def. of matching): the edges in M do not "touch" each other, one has to cover at least one vertex per edge of M. $\rightarrow$

$$|V_{\text{vc}}^{\text{min}}| \geq |M|\,. \tag{2}$$

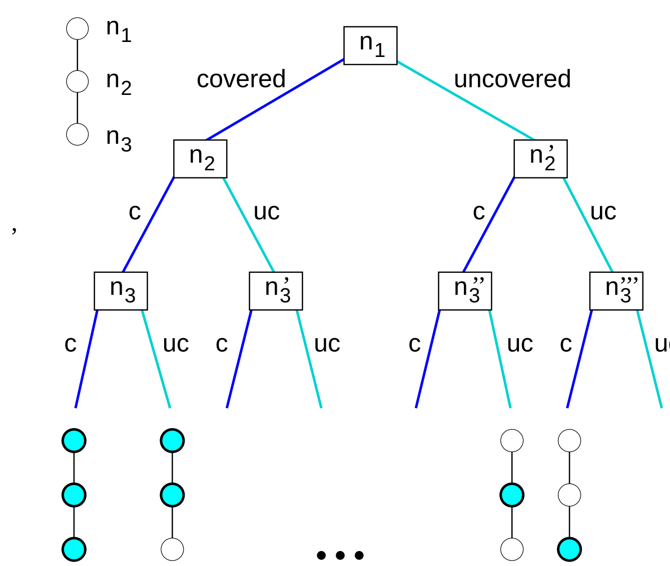
Combining Eqs (1) and (2) we get $|V_{\text{vc}}| = 2|M| \leq 2|V_{\text{vc}}^{\text{min}}|$. QED

3.2 Branch-and-bound algorithm

Finds exact minimum VC (optimization problem 2)

(Remark: if in algorithm a vertex i is (temporarily) covered, we say we put a covering mark on it. Vertices not decided yet (cov/uncov): free)

Basic idea: 2^N possible configurations $\in \{\text{cov}, \text{uncov}\}^N$
$\rightarrow$ binary configuration tree
$\rightarrow$ algorithms builds tree node by node (via backtracking) and determines smallest VC



$\rightarrow$ For sure exponential running time.
Speedup: omit subtrees if possible:

- No further descent if VC has been found.
- Cover neighbours of uncovered vertices.
- Bound. Store:
  - *best*: size of the smallest VC found so far (initially $best = N$).
  - X number of vertices covered so far
  - current degrees of free vertices d_i. Ordered $d_{o_1} \geq d_{o_2} \geq \ldots d_{o_{N'}}$

F :=*best*-X available number of covering marks
note: if only ONE best solution is to be obtained, one can use $F = best - X - 1$

$D := \sum_{l=1}^{F} d_{o_l}$ best one can achieve with F marks
if $D <$ # current uncovered edges **then** bound!

Example:
$F = 3$

i	d_i
5	7
23	6
12	6
33	6
2	5
⋮	⋮

```
algorithm branch-and-bound(G, best, X)
begin
    if all edges are covered then
    begin
        if X < best then best := X
        return;
    end;
    calculate F = best − X; D = \sum_{l=1}^{F} d_l;
    if D < number of uncovered edges then
        return;          comment bound;
    take one free vertex i with the largest current degree d_i;
    mark i as covered; comment left subtree
    X := X + 1;
    remove from E all edges {i, j} incident to i;
    branch-and-bound(G, best, X);
    reinsert all edges {i, j} which have been removed;
    X := X − 1;
    if (F ≥ number of current neighbors) then
    begin                comment right subtree;
        mark i as uncovered;
        for all neighbors j of i do
        begin
            mark j as covered; X := X + 1;
            remove from E all edges {j, k} incident to j;
        end;
        branch-and-bound(G, best, X);
        for all neighbors j of i do
            mark j as free; X := X − 1;
        reinsert all edges {j, k} which have been removed;
    end;
    mark i as free;
```

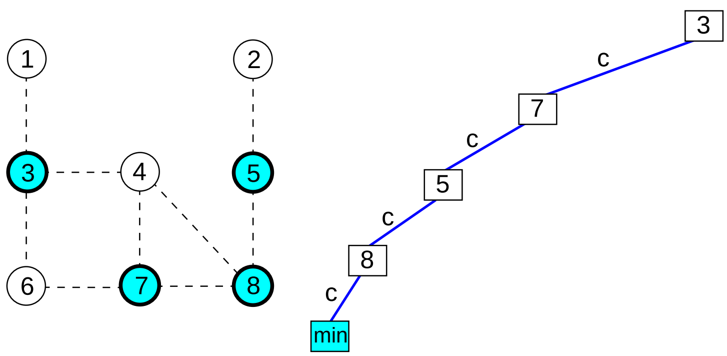
return;
end

first call: branch-and-bound(G, *best*,0).

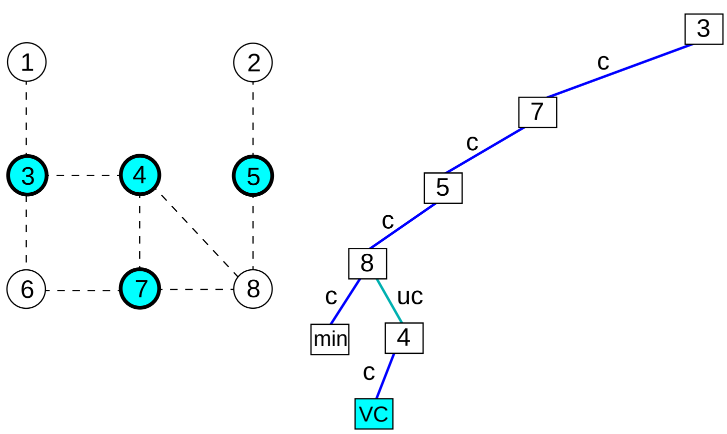
Example: Branch-and-bound algorithm

Graph from Ex. for heuristic.

First descent: exactly as the heuristics. → Fig. (graph and the corresponding current configuration tree): *best* := 4.

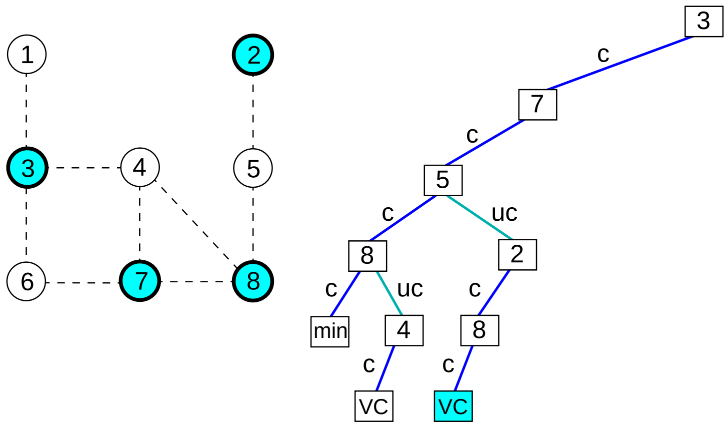


Algorithm → preceding level of the configuration tree. Vertex 8: *uncovered*. All its *uncovered* neighbours: *covered* (vertex 4) →



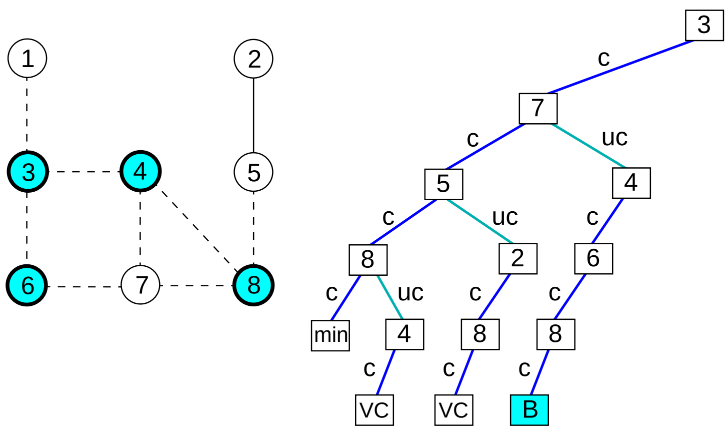
Next (recursive) call: Again full VC, but not smaller → backtracking.

Vertex 8 is *free* again, backtracking
Vertex 5:*uncovered* → its neighbours (2 and 8): *covered*



Next call: Again full VC, but not smaller → backtracking.

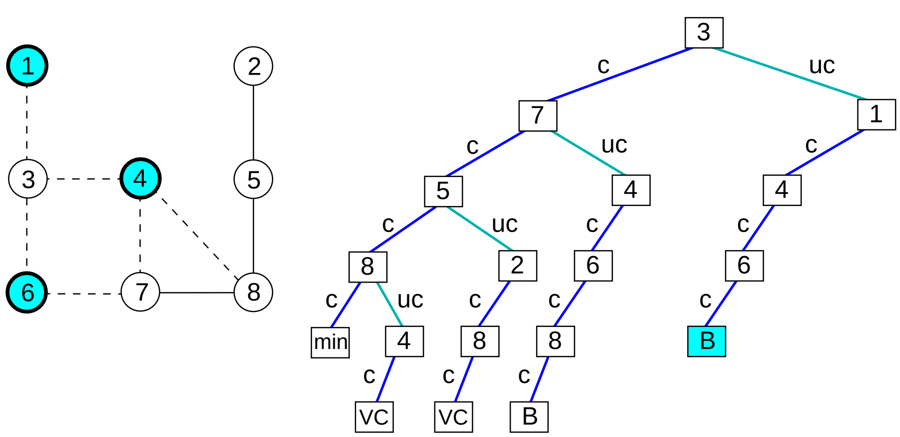
Vertex 5 is *free* again, backtracking
Vertex 7: *uncovered*, its neighbours, $(4, 6$ and $8)$: *covered*,



Next call: no cover yet (edge $\{2, 5\}$ is uncovered) → bound is evalated: $X = 4 \rightarrow F = best - X = 0 \rightarrow D = 0 < \#$ *uncovered* edges. → bound! →(no subtree) backtracking

Vertex 7 is *free* again, backtracking → top level

Vertex 3: *uncovered*, its neighbours, $(1, 4, 6)$: *covered*,



next call: no cover yet → bound evaluated: $X = 3 \rightarrow F = best - X = 4 - 3 = 1$: Vertex 8 has the highest current degree $d_8 = 2$, hence $D = 2$ $<$ # of uncovered edges is 3. → bound! →(no subtree) backtracking

→ algorithm finishes.

Note: configuration tree has 18 nodes, compared to 511 nodes (with $2^8 = 256$ leaves) of full configuration tree. □

Implementation : for fast access the F vertices of largest current degree (sublinear N treatment) →
two arrays v_1, v_2 of sets of vertices indexed by the current degrees.
v_1: top F *free* vertices
v_2: other *free* vertices
also store for each vertex: pointer to current set
insert/remove when *free* ↔ *covered*,*uncovered*
also lowest entry v_1 ↔ top entry v_2

Algorithm for optimization problem 1:
$\tilde{X} = |V_{\rm vc}|$ is given.
best: smallest number of uncovered edges (i. e., the energy) so far.
$F = \tilde{X} - X$ additional vertices coverable.
Again $D = \sum_{l=1}^{F} d_{O_l}$: sum of highest degrees.
If $best \leq$ (current # of uncovered edges)-D → bound !
(note: NO automatic covering of neighbors!)
stop if $best = 0$