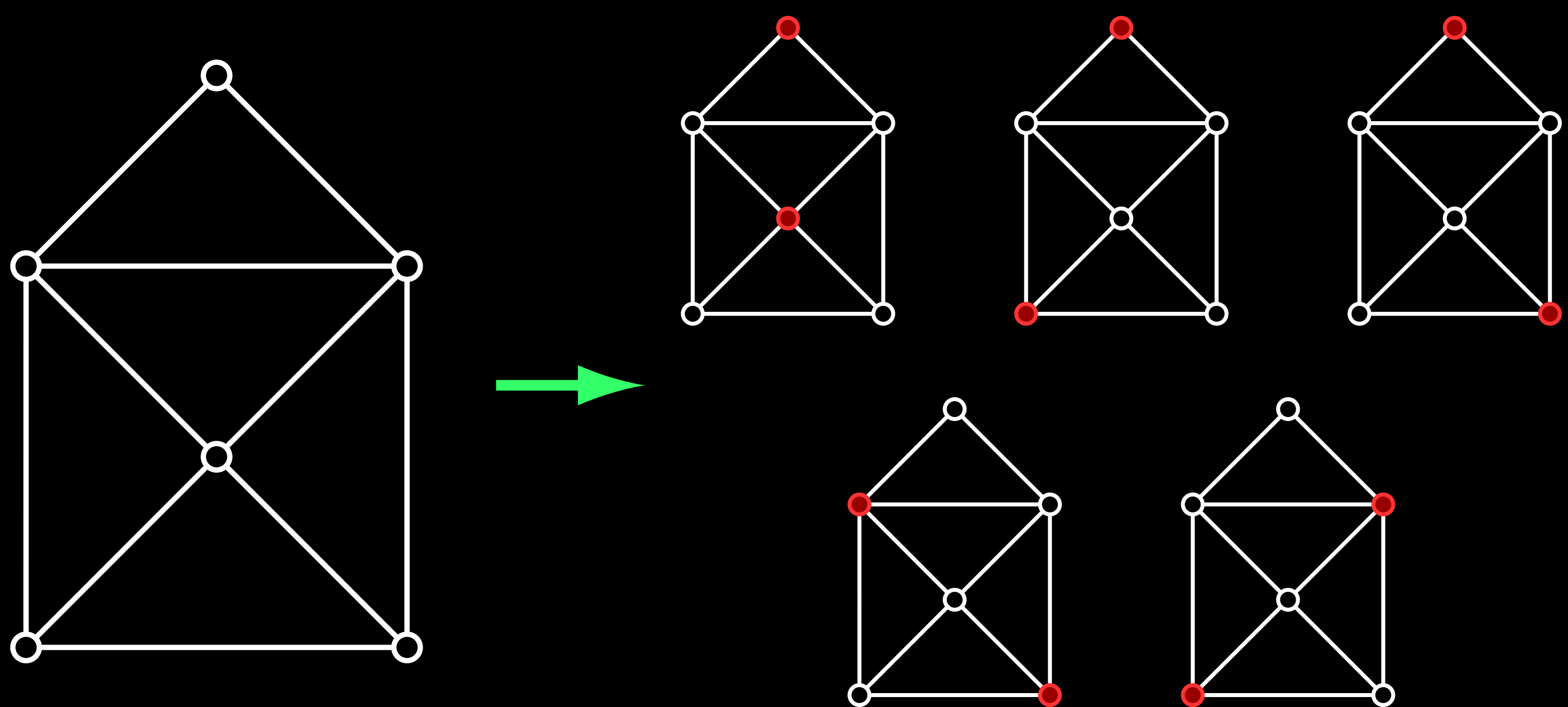
All Maximal Independent Sets and Dynamic Dominance for Sparse Graphs

David Eppstein

Univ. of California, Irvine
Donald Bren School of Information and Computer Sciences

Problem: list all maximal independent sets of an undirected graph

(equivalently, list all cliques in the graph complement)



Previously known results

$O(3^{n/3})$

matches lower bound on max possible output size

[Moon, Moser 1965; Lawler 1976]

$O(mn)$ per generated independent set

[Tsukiyama, Ide, Ariyoshi, Shirakawa 1977; Johnson, Yannakakis, Papadimitriou 1988]

$O(n^{2.376})$ per generated independent set

[Makino, Uno 2004]

Many additional heuristics

Many algorithms for special graph classes...

Main new results: faster generation for sparse graphs

$O(1)$ per generated independent set
for bounded degree graphs

$O(n)$ per generated independent set
for minor-closed graph families
including planar graphs

$O(n^{2-1/k})$ per generated independent set
for k-orientable graphs
(each subgraph with q vertices has ≤ kq edges;
e.g. planar graphs are 3-orientable)

Additional new results: dynamic domination

Maintain dynamic subset of graph vertices
Updates: insert or delete vertex into subset
Query: does subset dominate all remaining vertices?

$O(1)$ time per update for bounded degree graphs (trivial)

$O(1)$ time per update
for minor-closed graph families
including planar graphs

$O(n^{1-1/k})$ time per update
for k-orientable graphs

Main idea: Reverse Search [Avis, Fukuda 1992]

Given a family of objects to be enumerated

Find transformation *f*: object —> object
s.t. repeatedly applying *f* eventually leads to a canonical object

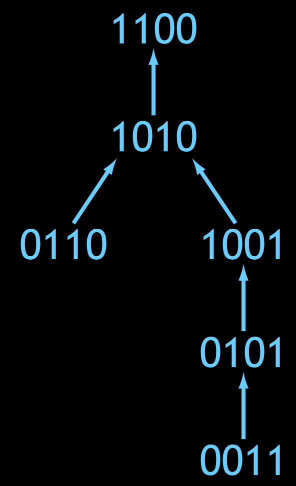
Form tree: nodes = objects parent = *f*(child)

Perform depth-first traversal of tree starting from canonical object

Example: n-bit words with k ones

f: replace leftmost 01 by 10

canonical object: $1^k 0^{n-k}$

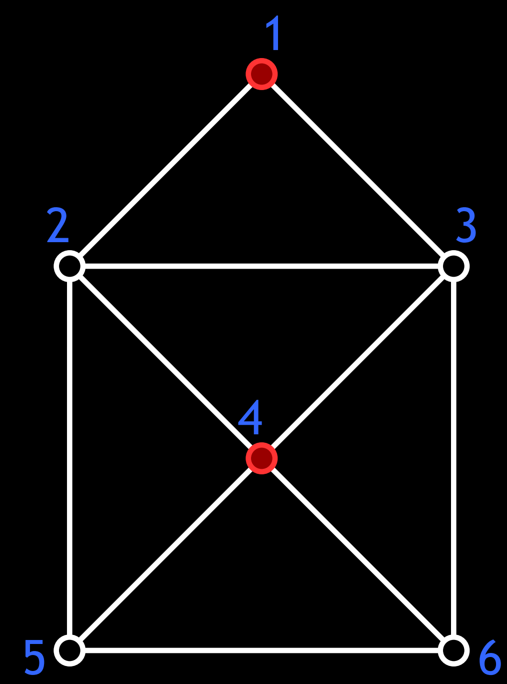


Canonical object for maximal independent sets: Lexicographically First Maximal Independent Set (LFMIS)

Number the vertices arbitrarily

Include a vertex in the LFMIS iff no lower-numbered neighbors are included

Can be constructed by simple linear-time greedy algorithm

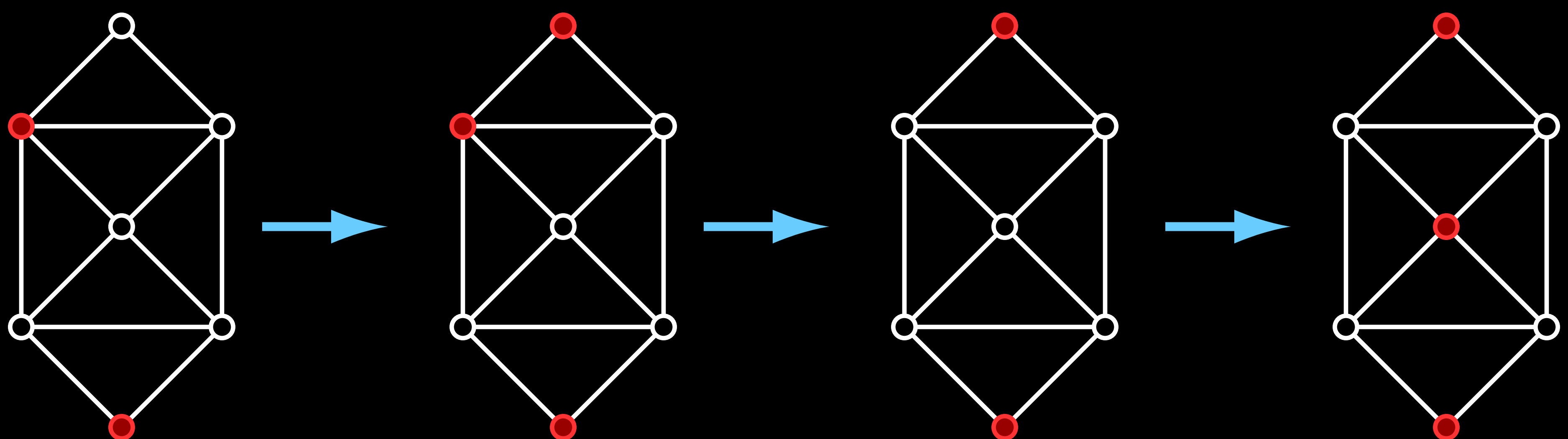


Transforming a maximal independent set towards the LFMIS

1. Find first missing LFMIS vertex v

2. Add v to set, make independent by removing neighbors of v

3. Complete to new maximal independent set greedily



Each transformation increases length of common prefix between set and LFMIS

Basic reverse search algorithm (recursive version)

```
def search(MIS):
    output MIS
    for each v in common prefix of MIS and LFMIS:
        S = higher-numbered neighbors of v
        for each nonempty independent subset I of S:
            child = (MIS union I) \ (neighbors of I)
            if child is a maximal independent set:
                search(child)

choose an ordering for graph's vertices
compute LFMIS
search(LFMIS)
```

Order by greedily removing min-degree vertices: $|S|=O(1)$ for sparse graphs
therefore, can list independent subsets of S in time $O(1)$

Nonrecursive version uses storage for $O(1)$ sets, avoids call stack

Speeding up the basic algorithm

Basic algorithm time: $O(n^2)$ per output

n potential children checked, O(n) per check

slow when many non-maximal children per maximal child

Bounded degree graphs:

Maintain dynamic set of triples (v,I,S)

leading only to children that are maximal independent sets

Each step of algorithm adds/removes O(1) triples from set

Children don't need checking, O(1) time per child

Minor-closed and more general sparse graphs:

Bottleneck is maximality test

An independent set is maximal iff it is dominating

Apply dynamic domination data structure

Dynamic domination for sparse graphs

Orient graph so outdegree = k = O(1)

Set degree threshhold

"high degree": degree ≥ $n^{1-1/k}$

Maintain easy dominance information

lowdom(v) = #dominating in-neighbors
+ #dominating low-degree out-neighbors

Group vertices according to their set of high-degree out-neighbors

Each high-degree vertex is adjacent to $O(n^{1-1/k})$ groups

Maintain dominance information about groups

#members with lowdom=0

#dominating high-degree out-neighbors

Set is dominating iff no group has
#undominated members > 0 and #dominating neighbors = 0

Dynamic domination for minor-closed graph families

Similar idea of grouping according to high-degree neighbors

#groups = O(# high degree)

With constant degree threshhold, subgraph of groups
is smaller by a constant factor than original graph

Continue grouping recursively to form hierarchical grouping structure

Each original vertex belongs to O(1) groups in hierarchical structure
so can maintain counts in all levels affected by update in O(1) time

Conclusions

Efficient reverse search for all maximal independent sets in sparse graphs

Complementary problem: cliques in dense graphs

(cliques in sparse, independent in dense are much easier)

Key subroutine: dynamic dominance data structure

Improved dynamic dominance for other sparse graph classes
would also improve independent set listing for those classes

Maybe can be extended to some non-sparse classes e.g. chordal graphs?