1 Introduction

Recall: Maximal Independent Sets

- A vertex set $S \subseteq V$ of a graph $G = (V, E)$ is an independent set in $G$ if there is no edge $uv \in E$ with $u, v \in S$.
- An independent set is maximal if it is not a subset of any other independent set.
- Examples:

![Graph with maximal independent sets]

Enumeration problem: Enumerate all maximal independent sets

<table>
<thead>
<tr>
<th>ENUM-MIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input:   graph $G$</td>
</tr>
<tr>
<td>Output:  all maximal independent sets of $G$</td>
</tr>
</tbody>
</table>

Maximal independent sets: $\{a, d\}, \{b\}, \{c\}$

Note: Let $v$ be a vertex of a graph $G$. Every maximal independent set contains a vertex from $N_G[v]$. 
Branching Algorithm for Enum-MIS

Algorithm \texttt{enum-mis}(G, I)

\textbf{Input} : A graph \(G = (V, E)\), an independent set \(I\) of \(G\).

\textbf{Output}: All maximal independent sets of \(G\) that are supersets of \(I\).

1 \(G' \leftarrow G - N_G[I]\)
2 \textbf{if} \(V(G') = \emptyset\) then \hfill // \(G'\) has no vertex
3 \textbf{else}
4 \quad Select \(v \in V(G')\) such that \(d_{G'}(v) = \delta(G')\) \hfill // \(v\) has min degree in \(G'\)
5 \quad \textbf{Run} \texttt{enum-mis}(G, I \cup \{u\}) \textbf{for each} \(u \in N_{G'}[v]\)

Running Time Analysis

Let us upper bound by \(L(n) = 2^\alpha n\) the number of leaves in any search tree of \texttt{enum-mis} for an instance with \(|V(G')| \leq n\).

We minimize \(\alpha\) subject to constraints obtained from the branching:

\[ L(n) \geq (d + 1) \cdot L(n - (d + 1)) \quad \text{for each integer } d \geq 0. \]

\[ \Leftrightarrow 2^{\alpha n} \geq d' \cdot 2^{\alpha (n-d')} \quad \text{for each integer } d' \geq 1. \]

\[ \Leftrightarrow 1 \geq d' \cdot 2^{\alpha (-d')} \quad \text{for each integer } d' \geq 1. \]

For fixed \(d'\), the smallest value for \(2^\alpha\) satisfying the constraint is \(d'^{1/d'}\). The function \(f(x) = x^{1/x}\) has its maximum value for \(x = e\) and for integer \(x\) the maximum value of \(f(x)\) is when \(x = 3\).

Therefore, the minimum value for \(2^\alpha\) for which all constraints hold is \(3^{1/3}\). We can thus set \(L(n) = 3^{n/3}\).

Since the height of the search trees is \(\leq |V(G')|\), we obtain:

\textbf{Theorem 1.} Algorithm \texttt{enum-mis} has running time \(O^*(3^{n/3}) \subseteq O(1.4423^n)\), where \(n = |V|\).

\textbf{Corollary 2.} A graph on \(n\) vertices has \(O(3^{n/3})\) maximal independent sets.

Running Time Lower Bound

\[ \cdots \]

\textbf{Theorem 3.} There is an infinite family of graphs with \(\Omega(3^{n/3})\) maximal independent sets.

2 Maximum Independent Set

\textbf{Maximum Independent Set}

\textbf{Input}: graph \(G\)

\textbf{Output}: A largest independent set of \(G\).
Branching Algorithm for Maximum Independent Set

Algorithm $\text{mis}(G)$

Input : A graph $G = (V,E)$.
Output: The size of a maximum i.s. of $G$.

1 if $\Delta(G) \leq 2$ then // $G$ has max degree $\leq 2$
2 \hspace{1em} return the size of a maximum i.s. of $G$ in polynomial time
3 else if $\exists v \in V : d(v) = 1$ then // $v$ has degree 1
4 \hspace{1em} return $1 + \text{mis}(G - N[v])$
5 else if $G$ is not connected then
6 \hspace{1em} Let $G_1$ be a connected component of $G$
7 \hspace{1em} return $\text{mis}(G_1) + \text{mis}(G - V(G_1))$
8 else
9 \hspace{1em} Select $v \in V$ s.t. $d(v) = \Delta(G)$ // $v$ has max degree
10 \hspace{1em} return $\max (1 + \text{mis}(G - N[v]), \text{mis}(G - v))$

Correctness

Line [1]

Lemma 4. If $v \in V$ has degree 1, then $G$ has a maximum independent set $I$ with $v \in I$.

Proof. Let $J$ be a maximum independent set of $G$. If $v \in J$ we are done because we can take $I = J$. If $v \notin J$, then $u \in J$, where $u$ is the neighbor of $v$, otherwise $J$ would not be maximum. Set $I = (J \setminus \{u\}) \cup \{v\}$. We have that $I$ is an independent set, and, since $|I| = |J|$, $I$ is a maximum independent set containing $v$.

2.1 Simple Analysis

Lemma 5 (Simple Analysis Lemma). Let

- $A$ be a branching algorithm
- $\alpha > 0$, $c \geq 0$ be constants

such that on input $I$, $A$ calls itself recursively on instances $I_1, \ldots, I_k$, but, besides the recursive calls, uses time $O(|I|^c)$, such that

$$(\forall i : 1 \leq i \leq k) \quad |I_i| \leq |I| - 1, \text{ and } 2^{\alpha|I|} + \ldots + 2^{\alpha|I_i|} \leq 2^{\alpha|I|}. \quad (1)$$

Then $A$ solves any instance $I$ in time $O(|I|^{c+1}) \cdot 2^{\alpha|I|}$.

Proof. By induction on $|I|$. W.l.o.g., suppose the hypotheses’ $O$ statements hide a constant factor $d \geq 0$, and for the base case assume that the algorithm returns the solution to an empty instance in time $d \leq d \cdot |I|^{c+1}2^{\alpha|I|}$.

Suppose the lemma holds for all instances of size at most $|I| - 1 \geq 0$, then the running time of algorithm $A$ on instance $I$ is

$$T_A(I) \leq d \cdot |I|^c + \sum_{i=1}^{k} T_A(I_i) \quad \text{(by definition)}$$

$$\leq d \cdot |I|^c + \sum_{i=1}^{k} d \cdot |I_i|^{c+1}2^{\alpha|I_i|} \quad \text{(by the inductive hypothesis)}$$

$$\leq d \cdot |I|^c + d \cdot (|I| - 1)^{c+1} \sum_{i=1}^{k} 2^{\alpha|I_i|} \quad \text{(by (1))}$$

$$\leq d \cdot |I|^c + d \cdot (|I| - 1)^{c+1}2^{\alpha|I|} \quad \text{(by (2))}$$

$$\leq d \cdot |I|^{c+1}2^{\alpha|I|}. \quad \text{(by induction)}$$

The final inequality uses that $\alpha \cdot |I| > 0$ and holds for any $c \geq 0$. 

\[ \square \]
Simple Analysis for mis

- At each node of the search tree: $O(n^2)$ time
- $G$ disconnected: (1) If $\alpha \cdot s < 1$, then $s < 1/\alpha$, and the algorithm solves $G_1$ in constant time (provided that $\alpha > 0$). We can view this rule as a simplification rule, removing $G_1$ and making one recursive call on $G - V(G_1)$. (2) If $\alpha \cdot (n - s) < 1$; similar as (1). (3) Otherwise, always satisfied since $2^x + 2^y \leq 2^{x+y}$ if $x, y \geq 1$.
- Branch on vertex of degree $d \geq 3$

Dividing all these terms by $2^{\alpha n}$, the constraints become

$$2^{-\alpha} + 2^{\alpha(-1-d)} \leq 1. \quad (5)$$

Compute optimum $\alpha$

The minimum $\alpha$ satisfying the constraints is obtained by solving a convex mathematical program minimizing $\alpha$ subject to the constraints (the constraint for $d = 3$ is sufficient as all other constraints are weaker).

Alternatively, set $x := 2^\alpha$, compute the unique positive real root of each of the *characteristic polynomials*

$$c_d(x) := x^{-1} + x^{-1-d} - 1,$$

and take the maximum of these roots [Kul99].

<table>
<thead>
<tr>
<th>$d$</th>
<th>$x$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.3803</td>
<td>0.4650</td>
</tr>
<tr>
<td>4</td>
<td>1.3248</td>
<td>0.4057</td>
</tr>
<tr>
<td>5</td>
<td>1.2852</td>
<td>0.3620</td>
</tr>
<tr>
<td>6</td>
<td>1.2555</td>
<td>0.3282</td>
</tr>
<tr>
<td>7</td>
<td>1.2321</td>
<td>0.3011</td>
</tr>
</tbody>
</table>

Simple Analysis: Result

- use the Simple Analysis Lemma with $c = 2$ and $\alpha = 0.464959$
- running time of Algorithm mis upper bounded by $O(n^3) \cdot 2^{0.464959 \cdot n} = O(2^{0.4650 \cdot n})$ or $O(1.3803^n)$

Lower bound

$$T(n) = T(n - 5) + T(n - 3)$$

- for this graph, $P_n^2$, the worst case running time is $1.1938 \ldots \cdot \text{poly}(n)$
- Run time of algo mis is $\Omega(1.1938^n)$
Worst-case running time — a mystery
What is the worst-case running time of Algorithm \texttt{mis}?

- lower bound $\Omega(1.1938^n)$
- upper bound $O(1.3803^n)$

2.2 Search Trees and Branching Numbers

Search Trees
Denote $\mu(I) := \alpha \cdot |I|$.

Example: execution of \texttt{mis} on a $P^2$

Branching number: Definition
Consider a constraint

\[ 2^{\mu(I) - a_1} + \ldots + 2^{\mu(I) - a_k} \leq 2^{\mu(I)}. \]

Its branching number is

\[ 2^{-a_1} + \ldots + 2^{-a_k}, \]

and is denoted by

\[ (a_1, \ldots, a_k). \]

Clearly, any constraint with branching number at most 1 is satisfied.

Branching numbers: Properties

Dominance For any $a_i, b_i$ such that $a_i \geq b_i$ for all $i$, $1 \leq i \leq k$,

\[ (a_1, \ldots, a_k) \leq (b_1, \ldots, b_k), \]

as $2^{-a_1} + \ldots + 2^{-a_k} \leq 2^{-b_1} + \ldots + 2^{-b_k}$.

In particular, for any $a, b > 0$,

either $(a, a) \leq (a, b)$ or $(b, b) \leq (a, b)$.

Balance If $0 < a \leq b$, then for any $\varepsilon$ such that $0 \leq \varepsilon \leq a$,

\[ (a, b) \leq (a - \varepsilon, b + \varepsilon) \]

by convexity of $2^x$. 
2.3 Measure & Conquer Analysis

- **Goal**
  - capture more structural changes when branching into subinstances

- **How?**
  - potential-function method, a.k.a., Measure & Conquer [FGK09]

- **Example: Algorithm mis**
  - advantage when degrees of vertices decrease

**Measure**

Instead of using the number of vertices, $n$, to track the progress of mis, let us use a measure $\mu$ of $G$.

**Definition 6.** A measure $\mu$ for a problem $P$ is a function from the set of all instances for $P$ to the set of non-negative reals.

Let us use the following measure for the analysis of mis on graphs of maximum degree at most 5:

$$\mu(G) = \sum_{i=0}^{5} \omega_i n_i,$$

where $n_i := |\{v \in V : d(v) = i\}|$.

**Measure & Conquer Analysis**

**Lemma 7 (Measure & Conquer Lemma).** Let

- $A$ be a branching algorithm
- $c \geq 0$ be a constant, and
- $\mu(\cdot), \eta(\cdot)$ be two measures for the instances of $A$,

such that on input $I$, $A$ calls itself recursively on instances $I_1, \ldots, I_k$, but, besides the recursive calls, uses time $O(\eta(I)^c)$, such that

$$\forall i \quad \eta(I_i) \leq \eta(I) - 1, \text{ and}$$

$$2^{\mu(I_1)} + \ldots + 2^{\mu(I_k)} \leq 2^{\mu(I)}.$$  \hfill (6)

Then $A$ solves any instance $I$ in time $O(\eta(I)^{c+1}) \cdot 2^{\mu(I)}$. \hfill (7)

**Analysis of mis for degree at most 5**

For $\mu(G) = \sum_{i=0}^{5} \omega_i n_i$ to be a valid measure, we constrain that

$$w_d \geq 0$$

for each $d \in \{0, \ldots, 5\}$

We also constrain that reducing the degree of a vertex does not increase the measure (useful for analysis of the degree-1 simplification rule and the branching rule):

$$-\omega_d + \omega_{d-1} \leq 0$$

for each $d \in \{1, \ldots, 5\}$

Lines 1-2 is a halting rule and we merely need that it takes polynomial time so that we can apply Lemma 7.

Lines 3-4 of mis need to satisfy (7).
The simplification rule removes \( v \) and its neighbor \( u \). We get a constraint for each possible degree of \( u \):

\[
2^{\mu(G)-\omega_1-\omega_d} \leq 2^{\mu(G)} \quad \text{for each } d \in \{1, \ldots, 5\}
\]

\[
\Leftrightarrow \quad 2^{-\omega_1-\omega_d} \leq 2^0 \quad \text{for each } d \in \{1, \ldots, 5\}
\]

\[
\Leftrightarrow \quad -\omega_1 - \omega_d \leq 0 \quad \text{for each } d \in \{1, \ldots, 5\}
\]

These constraints are always satisfied since \( \omega_d \geq 0 \) for each \( d \in \{0, \ldots, 5\} \). Note: the degrees of \( u \)'s other neighbors (if any) decrease, but this degree change does not increase the measure.

For lines 5–7 of mis we consider two cases.

If \( \mu(G_1) < 1 \) (or \( \mu(G - V(G_1)) < 1 \), which is handled similarly), then we view this rule as a simplification rule, which takes polynomial time to compute \( \text{mis}(G_1) \), and then makes a recursive call \( \text{mis}(G - V(G_1)) \). To ensure that instances with measure < 1 can be solved in polynomial time, we constrain that

\[
w_d > 0 \quad \text{for each } d \in \{3, 4, 5\}
\]

and this will be implied by other constraints.

Otherwise, \( \mu(G_1) \geq 1 \) and \( \mu(G - V(G_1)) \geq 1 \), and we need to satisfy (7). Since \( \mu(G) = \mu(G_1) + \mu(G - V(G_1)) \), the constraints

\[
2^{\mu(G_1)} + 2^{\mu(G - V(G_1))} \leq 2^{\mu(G)}
\]

are always satisfied since the slope of the function \( 2^x \) is at least 1 when \( x \geq 1 \). (I.e., we get no new constraints on \( \omega_1, \ldots, \omega_5 \).)

Lines 8–10 of mis need to satisfy (7). We know that in \( G - N[v] \), some vertex of \( N^2[v] \) has its degree decreased (unless \( G \) has at most 6 vertices, which can be solved in constant time). Define

\[
(\forall d : 2 \leq d \leq 5) \quad h_d := \min_{2 \leq i \leq d} \{w_i - w_{i-1}\}
\]

We obtain the following constraints:

\[
2^{\mu(G)-w_d-\sum_{i=2}^d p_i (w_i - w_{i-1})} + 2^{\mu(G)-w_d-\sum_{i=2}^d p_i w_i - h_d} \leq 2^{\mu(G)}
\]

\[
\Leftrightarrow \quad 2^{-w_d-\sum_{i=2}^d p_i (w_i - w_{i-1})} + 2^{-w_d-\sum_{i=2}^d p_i w_i - h_d} \leq 1
\]

for all \( d, 3 \leq d \leq 5 \) (degree of \( v \)), and all \( p_i, 2 \leq i \leq d \), such that \( \sum_{i=2}^d p_i = d \) (number of neighbors of degree \( i \)).

Applying the lemma

Our constraints

\[
w_d \geq 0
\]

\[
-\omega_d + \omega_{d-1} \leq 0
\]

\[
2^{-w_d-\sum_{i=2}^d p_i (w_i - w_{i-1})} + 2^{-w_d-\sum_{i=2}^d p_i w_i - h_d} \leq 1
\]

are satisfied by the following values:

<table>
<thead>
<tr>
<th>( i )</th>
<th>( w_i )</th>
<th>( h_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>3</td>
<td>0.35</td>
<td>0.10</td>
</tr>
<tr>
<td>4</td>
<td>0.38</td>
<td>0.03</td>
</tr>
<tr>
<td>5</td>
<td>0.40</td>
<td>0.02</td>
</tr>
</tbody>
</table>

These values for \( w_i \) satisfy all the constraints and \( \mu(G) \leq 2n/5 \) for any graph of max degree \( \leq 5 \). Taking \( c = 2 \) and \( \eta(G) = n \), the Measure & Conquer Lemma shows that mis has run time \( O(n^3)2^{2n/5} = O(1.3196^n) \) on graphs of max degree \( \leq 5 \).
2.4 Optimizing the measure

Compute optimal weights

- By convex programming [GaspersS09]

All constraints are already convex, except conditions for $h_d$

$$h_d := \min_{2 \leq i \leq d} \{w_i - w_{i-1}\}$$

$$2 \leq d \leq 5$$

Use existing convex programming solvers to find optimum weights.

Convex program in AMPL

```AMPL
param maxd integer = 5;
set DEGREES := 0..maxd;
var W (DEGREES) >= 0;  # weight for vertices according to their degrees
var g (DEGREES) >= 0;  # weight for degree reductions from deg i
var h (DEGREES) >= 0;  # weight for degree reductions from deg <= i
var Wmax;  # maximum weight of W[d]
minimize Obj: Wmax;  # minimize the maximum weight
subject to MaxWeight {d in DEGREES}:
   Wmax >= W[d];
subject to gNotation {d in DEGREES : 2 <= d}:
   g[d] <= W[d] - W[d-1];
subject to hNotation {d in DEGREES, i in DEGREES : 2 <= i <= d}:
   h[d] <= W[i] - W[i-1];
subject to Deg3 {p2 in 0..3, p3 in 0..3 : p2+p3=3}:
subject to Deg4 {p2 in 0..4, p3 in 0..4, p4 in 0..4 : p2+p3+p4=4}:
subject to Deg5 {p2 in 0..5, p3 in 0..5, p4 in 0..5, p5 in 0..5 : p2+p3+p4+p5=5}:
```

Optimal weights

<table>
<thead>
<tr>
<th>i</th>
<th>$w_i$</th>
<th>$h_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.206018</td>
<td>0.206018</td>
</tr>
<tr>
<td>3</td>
<td>0.324109</td>
<td>0.118091</td>
</tr>
<tr>
<td>4</td>
<td>0.356007</td>
<td>0.031898</td>
</tr>
<tr>
<td>5</td>
<td>0.358044</td>
<td>0.002037</td>
</tr>
</tbody>
</table>

- use the Measure & Conquer Lemma with $\mu(G) = \sum_{i=1}^{5} w_i n_i \leq 0.358044 \cdot n$, $c = 2$, and $\eta(G) = n$
- mis has running time $O(n^3)2^{0.358044\cdot n} = O(1.2817^n)$

2.5 Exponential Time Subroutines

Lemma 8 (Combine Analysis Lemma). Let

- $A$ be a branching algorithm and $B$ be an algorithm,
- $c \geq 0$ be a constant, and
- $\mu(\cdot), \mu'(\cdot), \eta(\cdot)$ be three measures for the instances of $A$ and $B$,
such that $\mu'(I) \leq \mu(I)$ for all instances $I$, and on input $I$, $A$ either solves $I$ by invoking $B$ with running time $O(\eta(I)^{c+1}) \cdot 2^{\mu'(I)}$, or calls itself recursively on instances $I_1, \ldots, I_k$, but, besides the recursive calls, uses time $O(\eta(I)^c)$, such that

$$\forall i \eta(I_i) \leq \eta(I) - 1 \text{, and}$$

$$2^{\mu'(I_1)} + \ldots + 2^{\mu'(I_k)} \leq 2^{\mu(I)}.$$

Then $A$ solves any instance $I$ in time $O(\eta(I)^{c+1}) \cdot 2^{\mu(I)}$.

Algorithm mis on general graphs

- use the Combine Analysis Lemma with $A = B = \text{mis}$, $c = 2$, $\mu(G) = 0.35805n$, $\mu'(G) = \sum_{i=1}^{5} w_i n_i$, and $\eta(G) = n$
- for every instance $G$, $\mu'(G) \leq \mu(G)$ because $\forall i, w_i \leq 0.35805$
- for each $d \geq 6$,

$$0.35805 \cdot (d + 1) \leq 1$$

Thus, Algorithm mis has running time $O(1.2817^n)$ for graphs of arbitrary degrees

2.6 Structures that arise rarely

Rare Configurations

- Branching on a local configuration $C$ does not influence overall running time if $C$ is selected only a constant number of times on the path from the root to a leaf of any search tree corresponding to the execution of the algorithm
- Can be proved formally by using measure

$$\mu'(I) := \begin{cases} 
\mu(I) + c & \text{if } C \text{ may be selected in the current subtree} \\
\mu(I) & \text{otherwise.}
\end{cases}$$

Avoid branching on regular instances in mis

else

Select $v \in V$ such that

(1) $v$ has maximum degree, and
(2) among all vertices satisfying (1), $v$ has a neighbor of minimum degree

return $\max (1 + \text{mis}(G - N[v]), \text{mis}(G - v))$

New measure:

$$\mu'(G) = \mu(G) + \sum_{d=3}^{5} [G \text{ has a } d\text{-regular subgraph}] \cdot C_d$$

where $C_d, 3 \leq d \leq 5$, are constants. The Iverson bracket $[F] = \begin{cases} 
1 & \text{if } F \text{ true} \\
0 & \text{otherwise}
\end{cases}$

Resulting Branching numbers

For each $d, 3 \leq d \leq 5$ and all $p_i, 2 \leq i \leq d$ such that $\sum_{i=2}^{d} p_i = d$ and $p_d \neq d$,

$$w_d + \sum_{i=2}^{d} p_i \cdot (w_i - w_{i-1}), w_d + \sum_{i=2}^{d} p_i \cdot w_i + h_d$$

All these branching numbers are at most 1 with the optimal set of weights
Thus, the modified Algorithm mis has running time $O(2^{0.3480n}) = O(1.2728^n)$.

Current fastest algorithm for MIS: $O(1.1996^n)$ \[XN17\]

## 3 Further Reading

- Chapter 2, *Branching* in \[FK10\]
- Chapter 6, *Measure & Conquer* in \[FK10\]
- Chapter 2, *Branching Algorithms* in \[Gas10\]

## References


