Greedy Algorithms

Basics of Greedy Algorithms

Greedy Algorithm

- Algorithms that takes a seq. of greedy decisions
- Pursues local, narrow-scope gains
- Mostly far from global optimum
- Sometimes(rarely) guarantees global optimum

```
do
        Select locally best-looking choice
until (valid solution is made)
```

A Typical Structure of Greedy Algorithm

Greedy(S):

◀ S: set of all elements

$X \leftarrow \emptyset$

while ($S \neq \emptyset$ **and** X is not yet a valid solution)

$x \leftarrow$ best-looking element in S

$S \leftarrow S - \{x\}$ ◀ Remove x in the set S

if (x can be added to X)

$X \leftarrow X \cup \{x\}$

if (X is a valid solution)

return X

else

return "failed!"

A constructive algorithm

Starts from an empty set
and grows to valid solution(s)

e.g., Prim, Kruskal, Dijkstra, …

cf. Iterative improvement algorithm

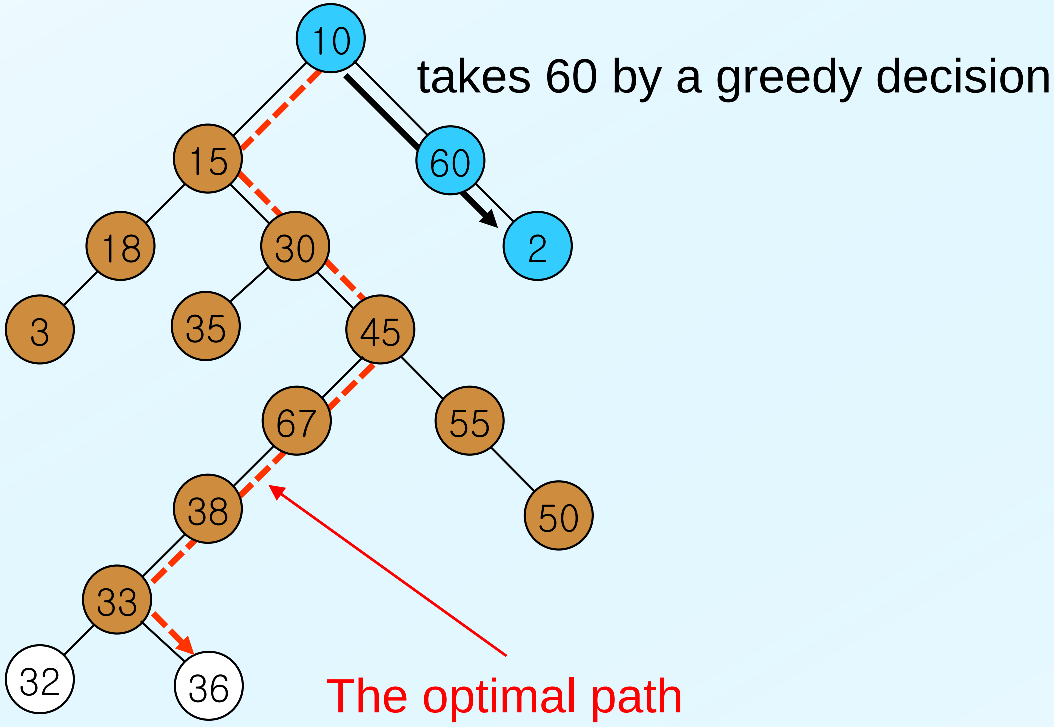
Starts from a valid solution
and keeps changing the solution

Example not Guaranting Optimum by a Greedy Algorithm

Given a binary tree

Starting at the root, find the maximal path length to a leaf.

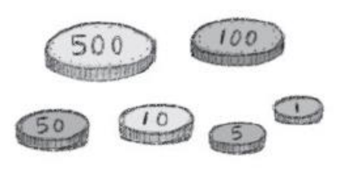
At a node, we can only see the numbers of its children.



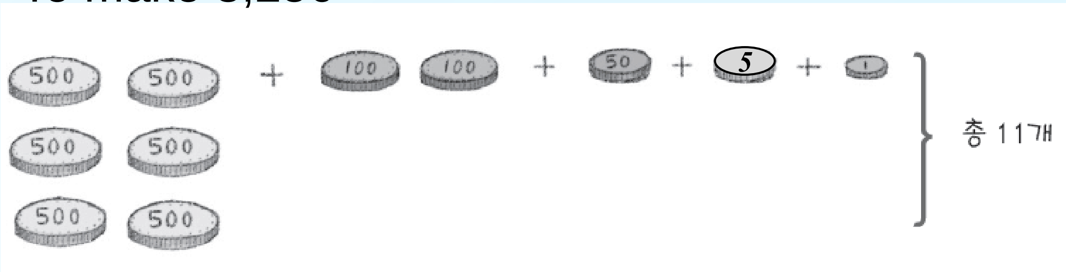
Example 2 not Guaranting Optimum by a Greedy Algorithm

Coin Exchange

Coin faces



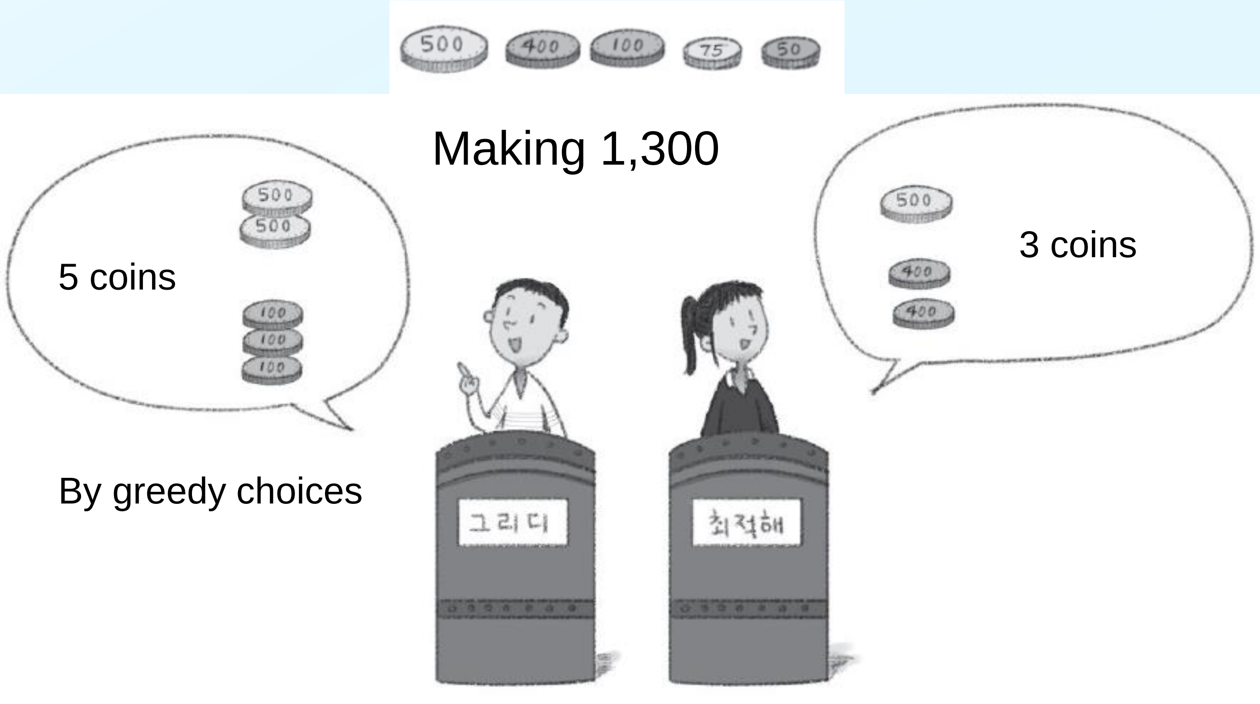
To make 3,256



Guarantees optimum
if every coin face is a multiple of the face immediately below

Do not guarantees optimum
if at least one coin face is not a multiple of the face immediately below
(example: next page)

Do not guarantees optimum
if at least one coin face is not a multiple of the face immediately below



Example Guaranting Optimum by a Greedy Algorithm

Prim algorithm and Kruskal algorithm
for minimum spanning trees

Prim(v):

Mark v as visited and include it in the m.s.t.

while (there are unvisited vertices)

Find a least-cost edge (x-u) from a visited vertex x
to an unvisited vertex u

Mark u as visited

Add the vertex u and the edge (x-u) to the m.s.t.

Greedy part

Example 2 Guaranting Optimum by a Greedy Algorithm

Room Assignment

- There is one meeting room
- Multiple teams try to make reservation
  - Have to provide the pair (starting time, ending time)
- Want to schedule the room to accept the maximum number of meetings

- Greedy ideas based on:
  - The order of durations: reserve w/ the shortest duration first
  - The starting time: reserve w/ the earliest starting time first
  - The ending time: reserve w/ the earliest ending time first

직관과 배치되지만

Only this guarantees optimum

Try to prove

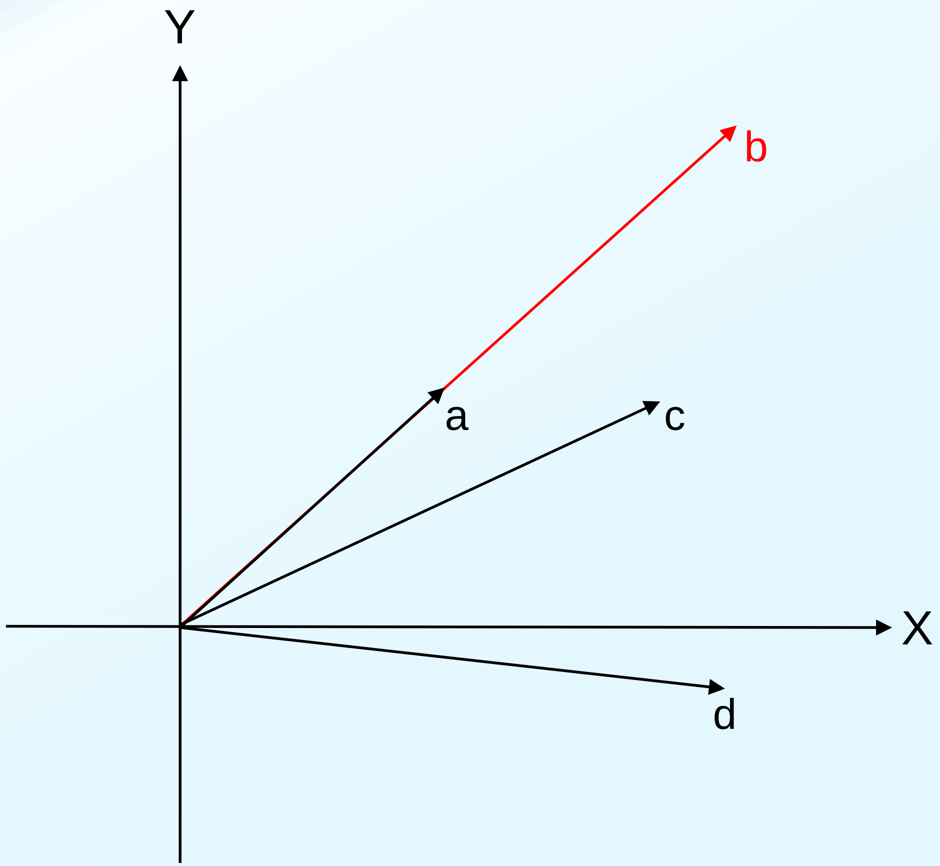
Matroid

A mathematical structure that abstracts and generalizes
the concept of linear independence in vector spaces

Linear algebra, graph theory, matching theory, field extension, routing theory 등에서 공통적으로 나오는 자연스러운 독립(independence) 개념들의 조합적 정수를 포착한다

-- 허준이, 필즈상 수상자

준비: Linear Independence(독립) in Vector Spaces



a와 b는 dependent(종속)
- a에 상수만 곱하면 b를 만들 수 있다

a와 c는 independent(독립)
- a로부터 어떻게 해도 (추가적인 vector를 쓰지 않고는) c를 만들 수 없다

{a, d}와 c는 dependent
- a와 d의 조합으로 c를 만들 수 있다

{a, c, d}는 dependent
- 둘의 조합으로 나머지 하나가 만들어지는 경우가 있다
- 예: a와 c의 조합으로 d를 만들 수 있다

{a, c}는 basis(기저)
- a와 c의 조합으로 모든 2차원 vector를 만들 수 있다
(equivalently, {a, c} spans $\mathcal{R}^2$)

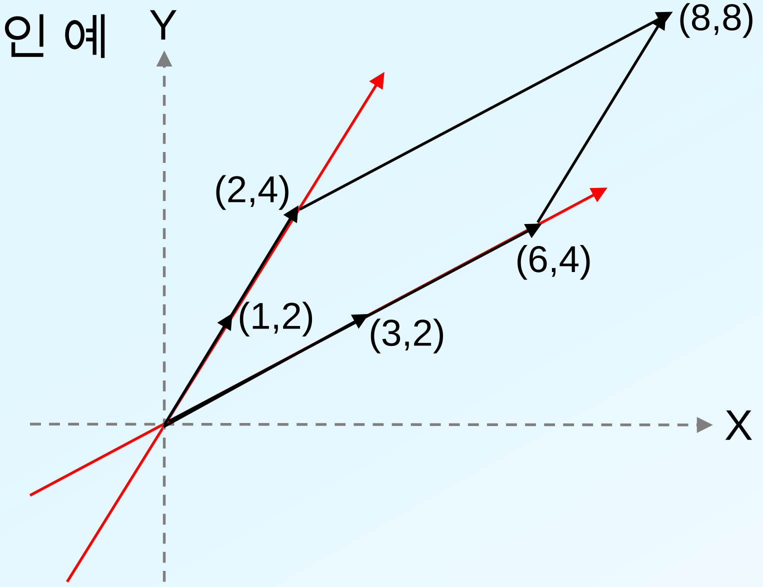
{a, d}도 basis, {b, c}도 basis, …

대표적 basis는 {(1,0), (0,1)} - 표준 기저(standard basis)라 한다

참고: Basis Change in Vector Spaces

개념을 돕기 위한 것인데, 오히려 혼란을 느끼는 사람은 무시해도 무방

$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$인 예



Basis changing matrix

$$\begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix} = a\begin{bmatrix} 1 \\ 2 \end{bmatrix} + b\begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} a+3b \\ 2a+2b \end{bmatrix}$$

Basis $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$, $\begin{bmatrix} 3 \\ 2 \end{bmatrix}$인 공간에서의 좌표 $\begin{bmatrix} a \\ b \end{bmatrix}$는

standard basis $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$인 공간에서 좌표 $\begin{bmatrix} a+3b \\ 2a+2b \end{bmatrix}$가 된다

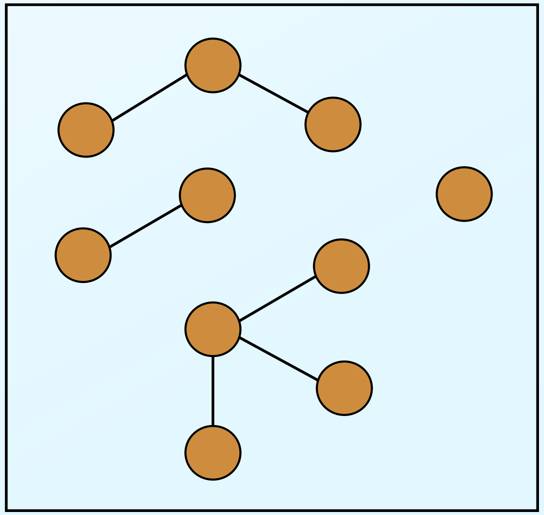
독립 개념의 전이: Cycle

그래프에서 cycle을 종속과 대응시킨다

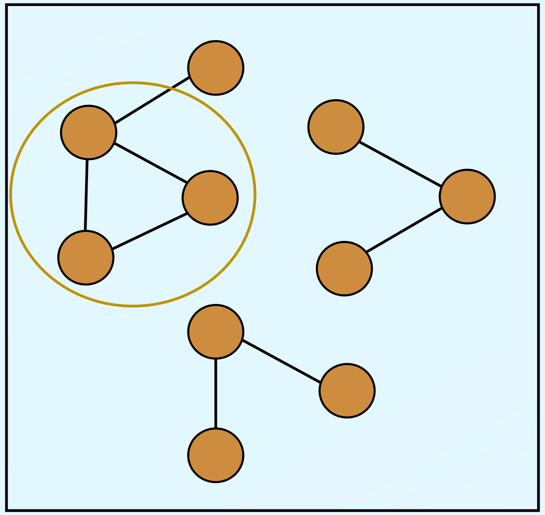
- 모든 forest(tree들의 집합)는 독립
- 모든 tree는 독립
- |V|개 이상의 edge로 이루어진 모든 집합은 종속

basis와 대응

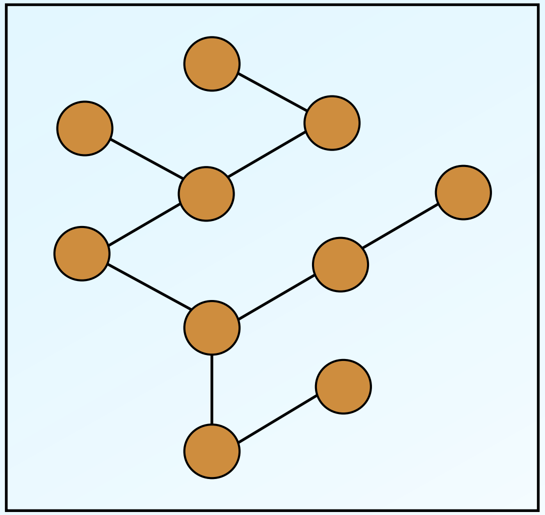
maximal independent set:
더 이상의 독립 집합으로 확장 불가



독립 (edge 집합)



종속 (edge 집합)



독립, maximal

Circuit

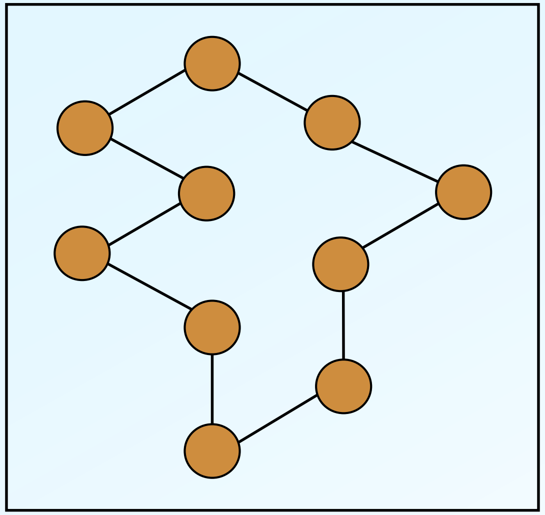
이것 때문에 오히려 혼란을 느끼는 사람은 그냥 무시할 것

- 종속 상태를 유지하는 최소한의 집합
- 어떤 하나를 제거하든 독립이 되는 집합
- Minimal dependent set

circuit

minimal dependent set:
더 이상의 종속 집합으로 축소 불가

그래프에서 circuit과 대응되는 것은 simple cycle



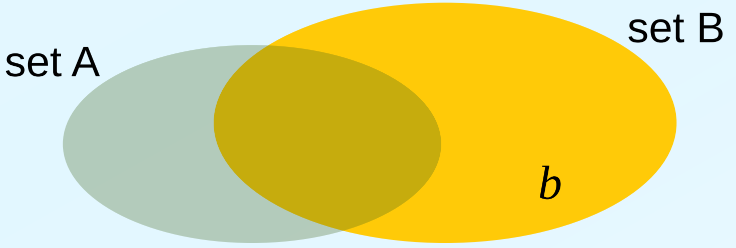
종속, minimal

Matroid

A Mathematical structure

If a problem has a matroid structure, a greedy algorithm guarantees optimum.

Definition 1: Matroid



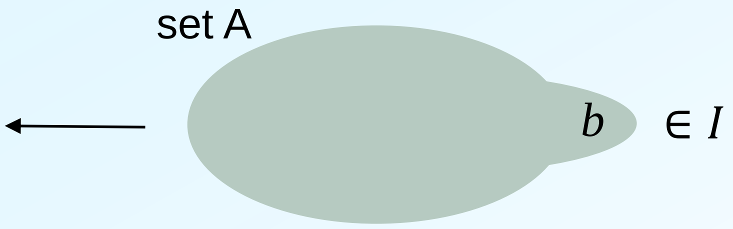
S: a finite set, I: a set of subsets of S ($I \subseteq 2^S$)
I is a **matroid** if it satisfies:

1.(heredity) If $A \in I$ and $B \subseteq A$, then $B \in I$

2.(extension or exchange property)

If $A, B \in I$ and $|A| < |B|$,

then there exists $b \in B - A$ such that $A \cup \{b\} \in I$.



✓ Invented by Hassler Whitney(1935), also by Takeo Nakasawa(1935~1938), independently

Simple Example 1

$$S = \{a, b, c, d\}, \qquad I = \{\emptyset, \{a\}, \{b\}, \{c\}, \{d\}\}$$

I is the set of all the subsets with one or fewer elements

Is I a matroid?

1. Heredity: Okay!
2. Extension property: Okay! $\longleftarrow$ $|\emptyset| < |\{b\}|$

I is a matroid!

Simple Example 2

$S = \{a, b, c, d\}$

$I = \{\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\}\}$

I is the set of all the subsets with two or fewer elements

Is I a matroid?

1. Heredity: Okay!
2. Extension property: Okay!

I is a matroid!

Simple Example 3

$$S = \{a, b, c, d\}$$

$$I = \{\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{c, d\}\}$$

I is the set of all the subsets with two or fewer elements except {*b*,*c*} and {*b*,*d*}

Is I a matroid?

1. Heredity: Okay!
2. Extension property: Not okay! $\longleftarrow$ $|\{b\}| < |\{c, d\}|$

I is not a matroid!

Extension of Matroid

Definition: Extension

In a matroid $I \subseteq 2^S$ and $A \in I$,

if $A \cup \{x\} \in I$ for an $x \in S, x \notin A$, we say x extends A.

If we cannot extend A any more, we say A is a maximal set.

≡ maximal independent set

≡ basis

Theorem 1:

All the maximal sets in a matroid $I \subseteq 2^S$ have the same size.

<Proof> Assume for contradiction that there are two maximal sets *A*, *B* ∈ I s.t. |*A*| < |*B*|.

By property 2, there exists $b \in B - A$ such that $A \cup \{b\} \in I$.

That is, we can extend *A*. Contradiction to the given condition that *A* is maximal.

e.g.: In a graphic matroid (the set of all forests $F \subseteq 2^E$),

every maximal set is a spanning tree and of size |*V*|-1.

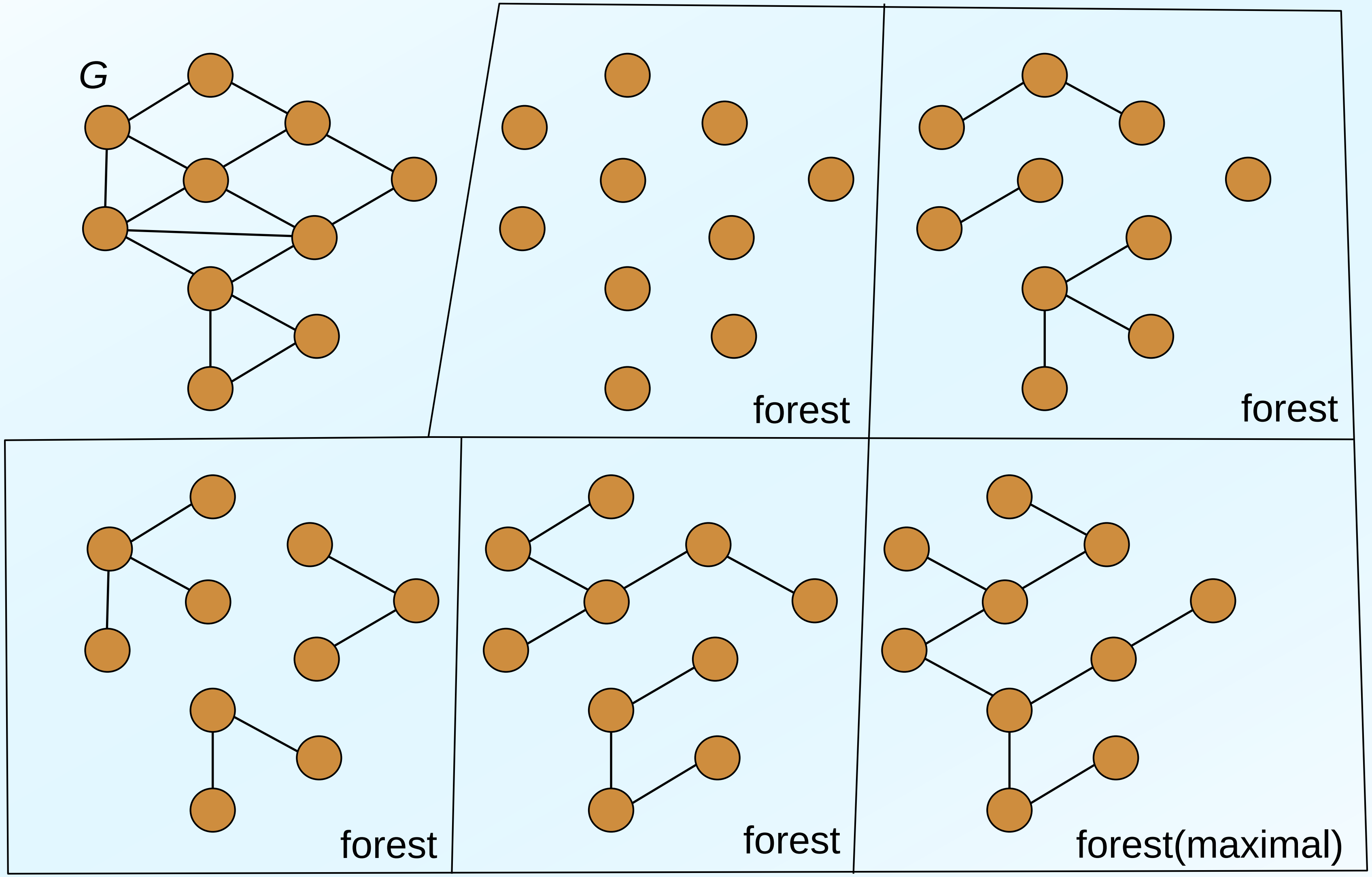
(# of edges = |*V*|-1)

Graphic Matroid

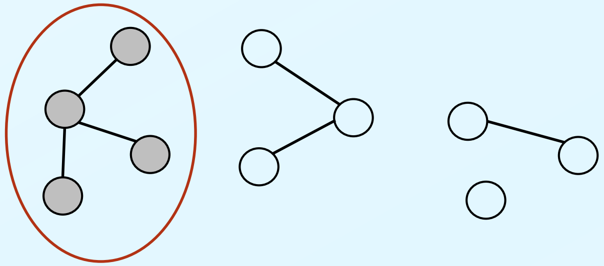
The set of all forests is a matroid

- Forest
  - A set of trees
  - Or, a set of edges with no cycle
- The set of all forests $F \subseteq 2^E$ out of a graph G=(V, E) is a matroid

Examples of Forests



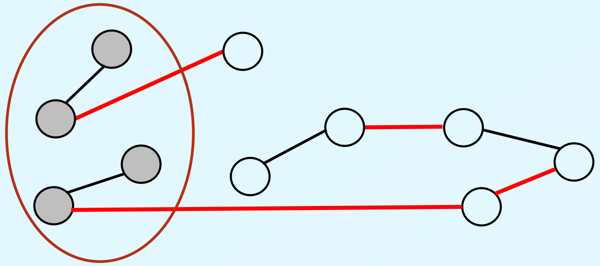
forest A (with 4 trees)



Theorem 2: (Graphic Matroid)

The set of all forests $F \subseteq 2^E$ of a graph $G = (V,E)$ is a matroid.

forest B (with 2 trees)



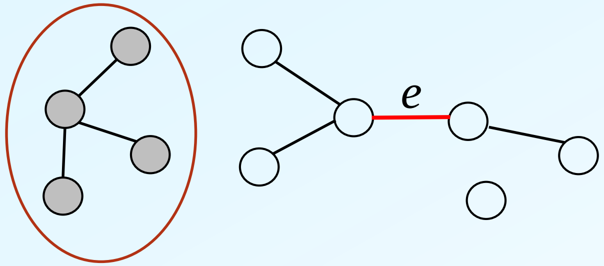
<Proof>

(1) (heredity) A subset of a forest is also a forest (trivial)

(2) (extension)

Consider two forests A and B (|A|<|B| WLOG).
Since A is not maximal, A has at least two separate trees.

forest A∪{e}

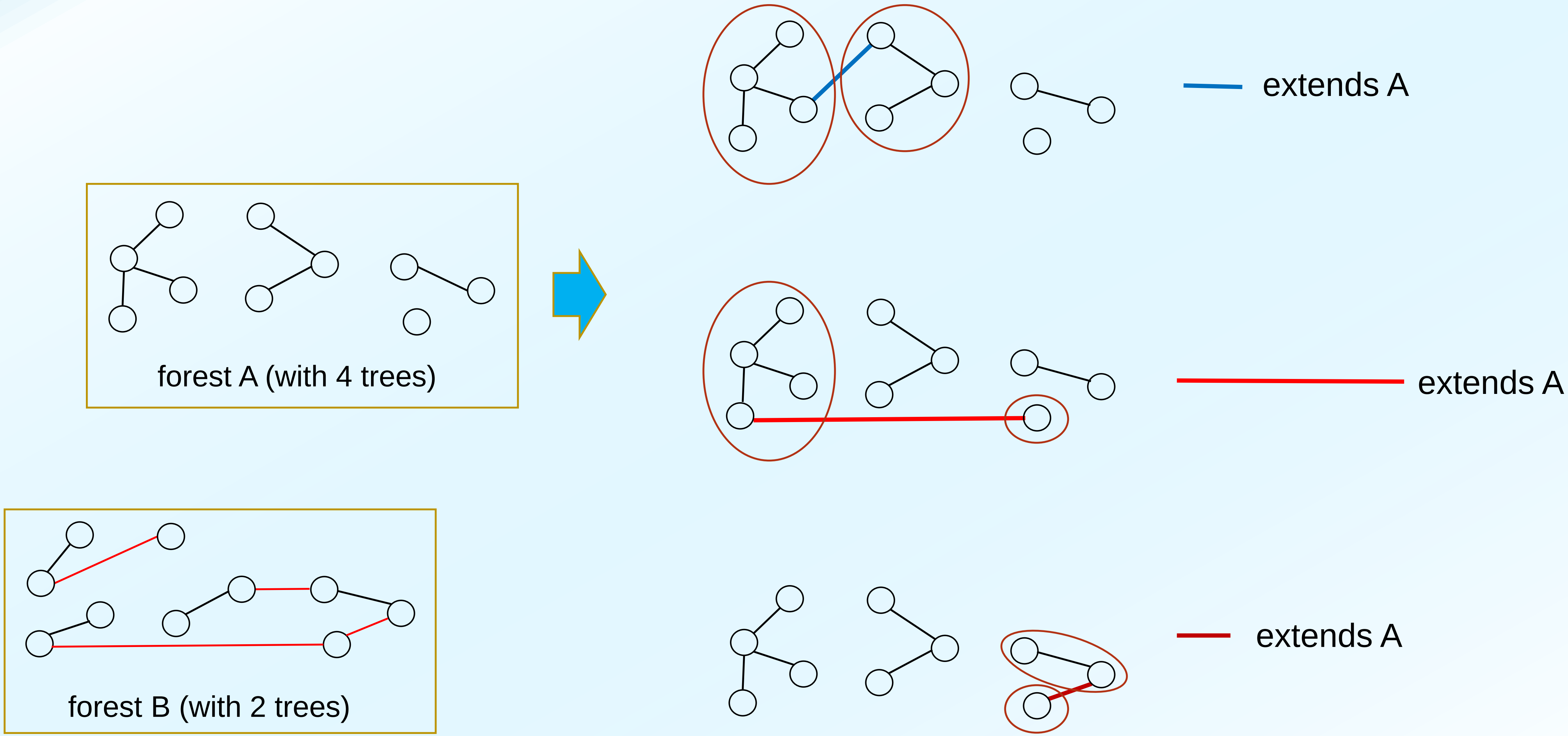


Let T be an arbitrary tree in A.
For the vertices of T, B has the same or fewer # of edges than A connecting vertices inside T. (If not, B has a cycle)
The same property holds for all the other trees in A.

Since B has more edges than A, B has at least one edge connecting two different trees in A.

Adding any of such edges to A does not make a cycle and makes a new forest.(i.e., the edge extends A) ∎

보충 설명: B의 edge들 중 A를 extend하는 나머지 예들



참고: Extension is the Same as Exchange

S: a finite set, I: matroid, a set of subsets of S ($I \subseteq 2^S$)

Extension

$A, B \in I$ and $|A| < |B|$,

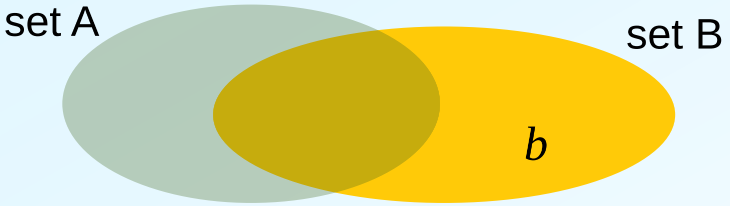
there exists $b \in B - A$ such that $A \cup \{b\} \in I$.

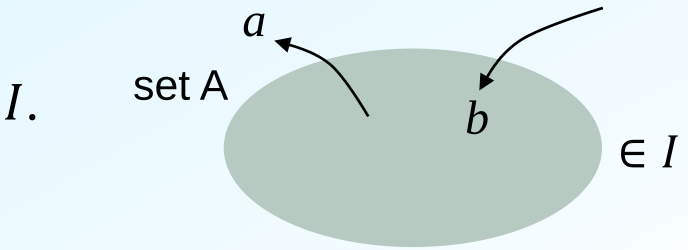
iff

Exchange

$A, B \in I$ and $A - B, B - A \neq \phi$,

for any $b \in B - A$, there exists $a \in A - B$ such that $A \cup \{b\} - \{a\} \in I$.





다른 정의

* basis: a maximal set

Definition 2: Matroid

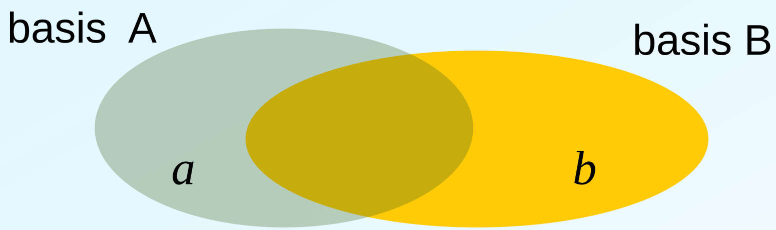
S: a finite set, $\mathcal{B} \subseteq 2^S$: a set of bases ← $\mathcal{B}$: a set of maximal independent set

$\mathcal{B}$ is a **matroid** if it satisfies:

1. $\mathcal{B} \neq \phi$
2. (basis exchange)

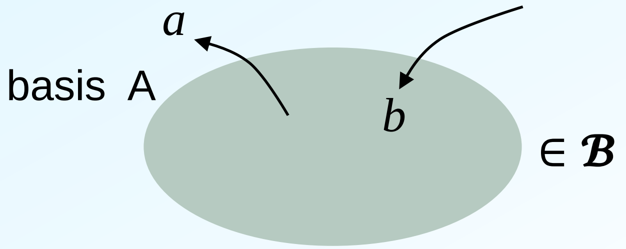
   For any bases $A, B \in \mathcal{B}$, $a \in A - B$,

   there exists $b \in B - A$ such that $A \cup \{b\} - \{a\} \in \mathcal{B}$.



* a basis(기저):

- a basis in linear algebra (vector 집합)
- a spanning tree(maximal forest) in a graph (edge 집합)



* 온전한 한 해는 basis이자 maximal independent set이다

Theorem 3: Graphic Matroid (by Definition 2)

The set of all maximal forests (=spanning trees) $\mathcal{T} \subseteq 2^E$ of a connected graph $G = (V, E)$ is a matroid.

<Proof>

(2) (basis exchange)

For any $A, B \in \mathcal{T}$, $e_1 \in A - B$,
there exists $e_2 \in B - A$ such that $A \cup \{e_2\} - \{e_1\} \in \mathcal{T}$.

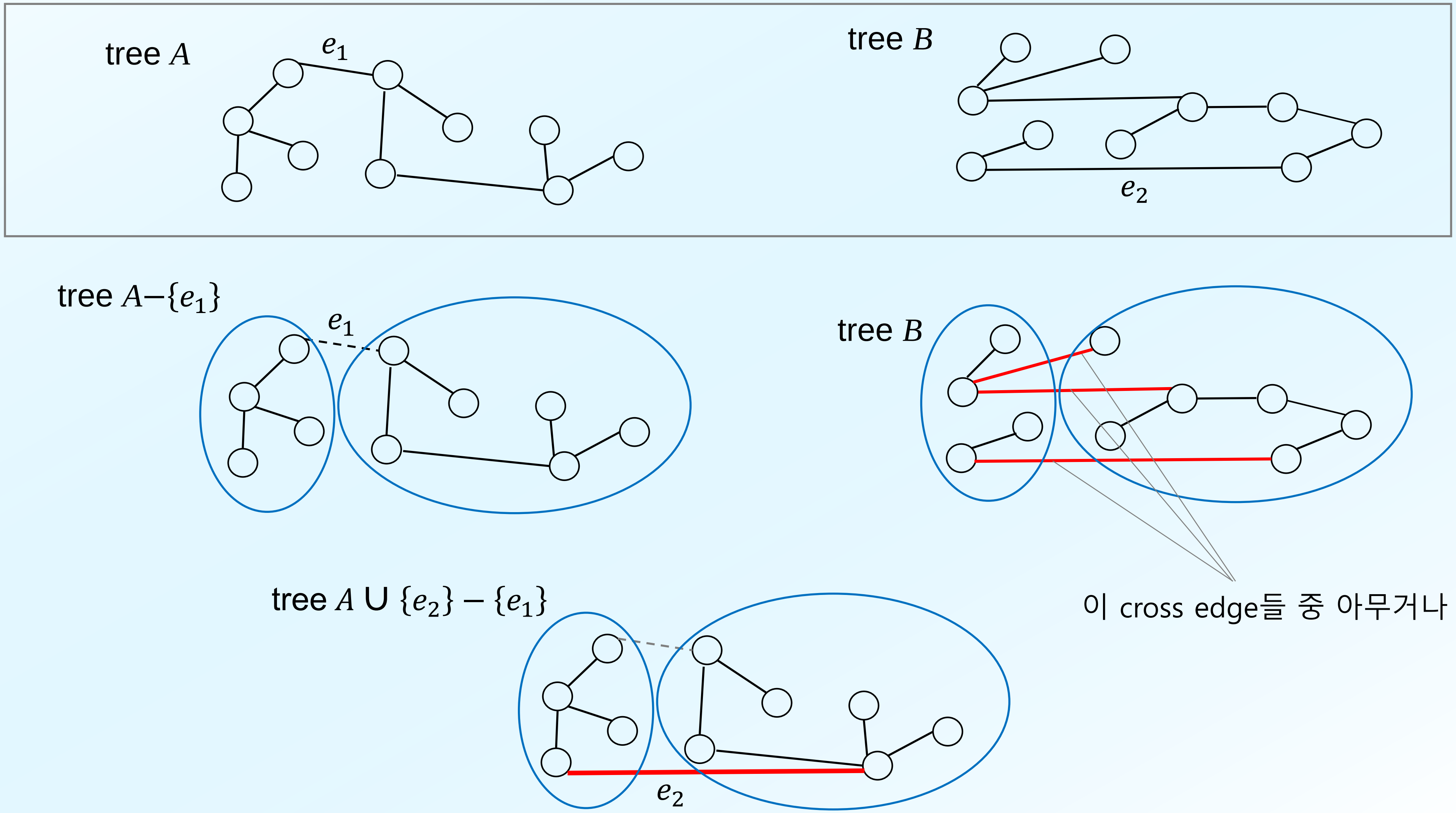
(직관적으로 설명하는 그림은 다음 페이지에)

<Proof>

(2) (basis exchange) (아래 직관적 그림으로)

For any $A, B \in \mathcal{T}$, $e_1 \in A - B$,

there exists $e_2 \in B - A$ such that $A \cup \{e_2\} - \{e_1\} \in \mathcal{T}$.



상상에 도움될 만한 것

이것 때문에 오히려 혼란을 느끼는 사람은 그냥 무시할 것

Definition 3: Circuit

A minimal dependent set in 2^S whose proper subsets are all independent

* Graphic matroid에서 circuit은 simple cycle과 일치

Weighted Matroid

In a matroid I from the set S of positive elements S,
we want to find a basis $A \in I$ that maximizes the sum of elements.

The greedy algorithm below guarantees an optimal solution.

Greedy(I, w[]):
▷ I : matroid, w[]: weight array
  $A = \emptyset$
  Sort all the elements of S in nonincreasing order of weights
  for each $x \in S$ (in nonincreasing order of weight)
    if $(A \cup \{x\} \in I)$
      $A \leftarrow A \cup \{x\}$
  return A

Proof of the Algorithm's Optimality

Greedy(I, w[]):
▷ I : matroid, w[]: weight array
    $A = \emptyset$
    Sort all the elements of S in nonincreasing order of weights
    for each $x \in S$ (in nonincreasing order of weight)
        if $(A \cup \{x\} \in I)$
            $A \leftarrow A \cup \{x\}$
    return A

최적해 X가 Greedy()로 구한 집합 A보다 가중치가 크다 가정하자.

알고리즘에서 집합 A는 더 이상 extension 되지 않을 때까지 가므로 maximal set이다.
가중치가 양이므로 최적해 X도 당연히 maximal set이다.

A의 원소들을 가중치 크기순으로 $\{a_1, a_2, \ldots, a_n\}$이라 하자.

마찬가지 방식으로 X의 원소들을 $\{x_1, x_2, \ldots, x_n\}$이라 하자.

$w(X) > w(A)$이므로 $w(x_i) > w(a_i)$인 i가 적어도 하나 존재한다. 그런 최초의 $i = k$라 하자.

$A_{k-1} = \{a_1, a_2, \ldots, a_{k-1}\} \in I$, $X_k = \{x_1, x_2, \ldots, x_k\} \in I$ 이다. (by heredity)

$|A_{k-1}| < |X_k|$이므로, $A_{k-1} \cup \{x_r\} \in I$인 $x_r \in X_k - A_{k-1}$이 존재한다.(by extension)

Since $w(x_r) \geq w(x_k) > w(a_k)$, 알고리즘 Greedy()는 a_k 대신 x_r을 선택한다.

∴ A보다 가중치가 큰 X가 존재하면 A는 알고리즘 Greedy()의 결과물이 될 수 없다.

∴ (대우) A가 알고리즘 Greedy()의 결과물이라면 A보다 가중치가 큰 X는 존재하지 않는다. ∎

Interesting Property

서로 다른 minimum spanning tree들도 가중치 집합은 동일하다!

Theorem 4:

If there exist two optimal solutions $A, B \in I$ in a weighted matroid I (the same weight sum, different subset), the sets of weight values are the same.

<Proof> (오류 수정)

두 최적해 A, B의 가중치 집합이 다르다 가정해보자.

$A - B$의 원소들을 가중치 크기순으로 $\{a_1, a_2, \dots, a_k\}$, $B - A$의 원소들을 가중치 크기순으로 $\{b_1, b_2, \dots, b_k\}$라 하자.

i번째 원소에서 최초로 $w(a_i) \neq w(b_i)$가 된다 하자. WLOG, let $w(a_i) < w(b_i)$.

By heredity, $(A \cap B) \cup \{a_1, a_2, \dots, a_{i-1}\} \in I$, $(A \cap B) \cup \{b_1, b_2, \dots, b_i\} \in I$.

Since $|(A \cap B) \cup \{a_1, a_2, \dots, a_{i-1}\}| < |(A \cap B) \cup \{b_1, b_2, \dots, b_i\}|$, by extension property,

there exists $b_i' \in \{b_1, b_2, \dots, b_i\}$ s.t. $(A \cap B) \cup \{a_1, a_2, \dots, a_{i-1}\} \cup \{b\} \in I$.

Let this set $(A \cap B) \cup \{a_1, a_2, \dots, a_{i-1}\} \cup \{b_i'\} = A'$.

Since $|A'| < |(A \cap B) \cup \{a_i, a_{i+1}, \dots, a_k\}|$, by extension property, there exists $a_{i+1}' \in \{a_i, a_{i+1}, \dots, a_k\}$ s.t. $A' \cup \{a_{i+1}'\} \in I$.

By repeating this way, we have $A' \cup \{a_{i+1}', \dots, a_k'\} \in I$, where $\{a_{i+1}', \dots, a_k'\} \subseteq \{a_i, a_{i+1}, \dots, a_k\}$.

Let this set $A' \cup \{a_{i+1}', \dots, a_k'\} = (A \cap B) \cup \{a_1, a_2, \dots, a_{i-1}\} \cup \{b_i'\} \cup \{a_{i+1}', \dots, a_k'\} = A''$.

Since the weight of b_i' is greater than any of $\{a_i, a_{i+1}, \dots, a_k\}$, $w(A) < w(A'')$;

Contradiction to the fact that A is optimal. ∎

Covering Both Maximization and Minimization

In a matroid I from the set S of positive elements S,
we want to find a maximal set $A \in I$ that maximizes(minimizes) the sum of elements.

The greedy algorithm below guarantees an optimal solution.

Greedy(I, w[]):
▷ I : matroid, w[]: weight array
    $A = \emptyset$
    Sort all the elements of S in nonincreasing(nondecreasing) order of weights
    for each $x \in S$ (in nonincreasing(nondecreasing) order of weight)
        if ($A \cup \{x\} \in I$)
            $A \leftarrow A \cup \{x\}$
    return A

Interesting Property

(Unimodality) In the solution space of a weighted matroid,
there exists only one peak possibly with more than one optimal solution.

Which of the Following are Possible?

Solution space of a weighted matroid

✓ Only (a) is possible!