データ構造とアルゴリズム 第14週

掛下　哲郎
kake@is.saga-u.ac.jp

講義スケジュール

データ構造 ↑ ↓ アルゴリズム

週	講義計画
1－2	導入
3	探索問題
4－5	基本的なデータ構造
6	動的探索問題とデータ構造
7	アルゴリズム演習（第1回）
8－9	データの整列
10－11	グラフアルゴリズム
12	文字列照合のアルゴリズム
13	アルゴリズム演習（第2回）
14	アルゴリズムの設計手法
15	計算困難な問題への対応

- 再帰（recursion）
- 分割統治法（divide and conquer）
- グリーディ法（greedy method，欲張り法，貪欲法）
- 動的計画法（dynamic programming）
- 分枝限定法（branch and bound）

様々なアルゴリズムを設計する際に使われる共通の戦略

再帰（recursion）

- 再帰手続き/関数
  - 自分自身を呼び出す手続き/関数
- 数学的帰納法と同じ原理
  - 帰納法
    - P(1)が成り立つ．P(k)が成り立てばP(k+1)も成り立つ
  - 再帰

    バージョンA：F(1)は解ける．F(k)を利用してF(k+1)を解く

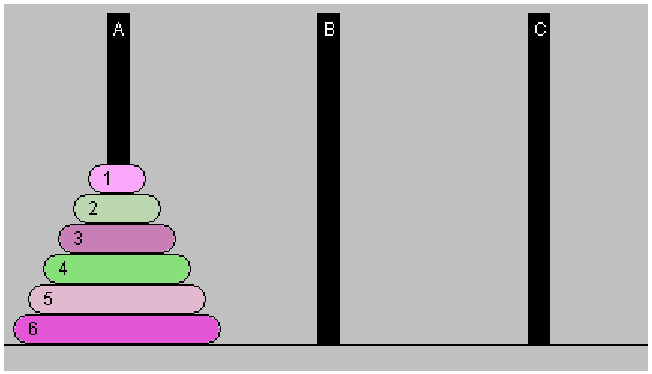
    バージョンB：F(1)は解ける． F(1)，・・，F(k)を利用してF(k+1)を解く
- 再帰的に定義できる問題では，アルゴリズムを単純化できる

再帰を用いたアルゴリズム例

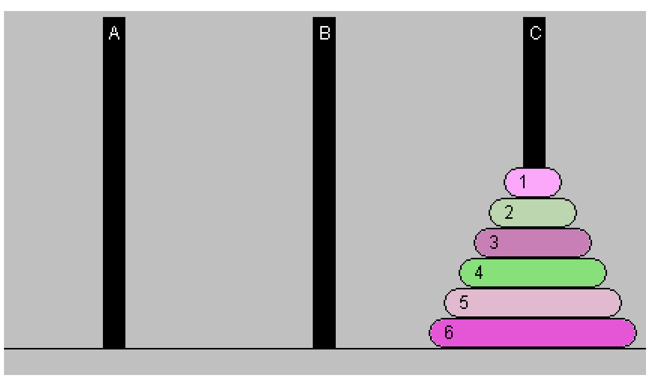
- ソーティング
  - クイックソート
  - マージソート
- 木構造の探索
  - 幅優先探索，深さ優先探索
  - Pre Order, In Order, Post Order
  - Minimax法
- 式の値の計算
- ハノイの塔

ハノイの塔（Tower of Hanoi）

開始



終了



問題

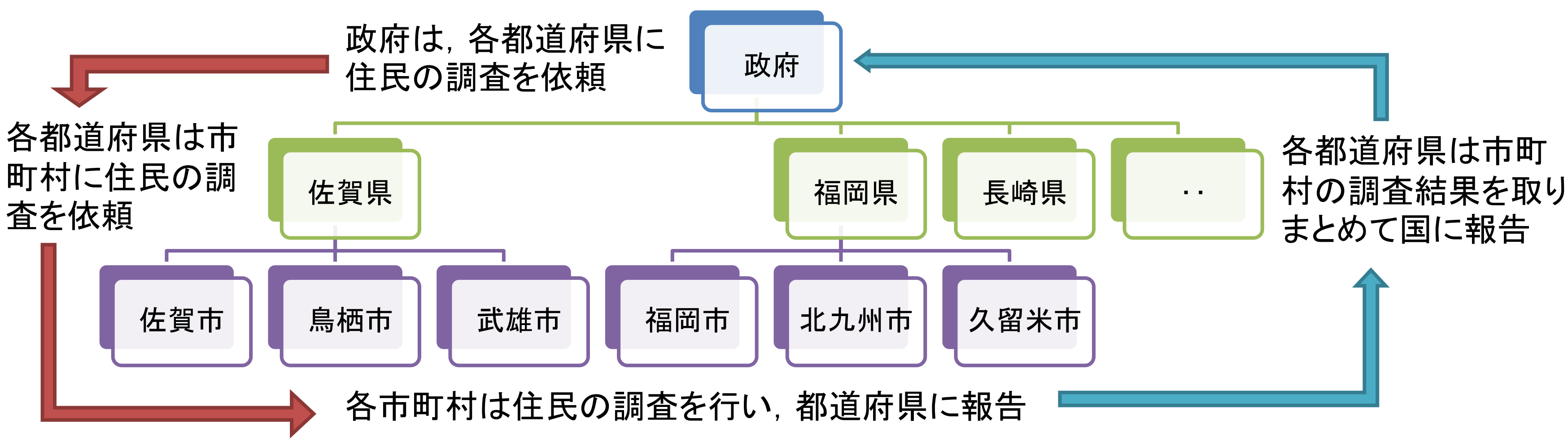
- A, B, Cの3本の棒と，中心に穴が開いたn枚の円盤がある．すべての円盤がAに積まれた状態から円盤を1枚ずつ動かしてCに移動せよ．ただし，小さい円盤の上に，より大きい円盤を置いてはいけない．

考え方

- n-1枚の円盤をAからBに移動する．
- 一番大きな円盤をAからCに移動する．
- n-1枚の円盤をBからCに移動する．

分割統治法（divide and conquer）

- 問題をそのまま扱うのではなく、
  - 分割：部分問題にばらす
  - 統治：部分問題をそれぞれ解く
  - 統合：部分問題の解を組み合わせる
- 例：日本全国の国民を調査する（国勢調査）



分割統治法の手続きF

1. 与えられた問題Pを部分問題P_1, P_2, ..., P_kに分割する.

分割

2. 各部分問題P_iに対して以下の処理を行う.

   2-1 P_i のサイズが小さいならば，直接解く.

   2-2 そうでなければ，P_iを分割して手続きFを再帰的に適用する.

統治

3. P_1, P_2, ..., P_kの解を利用してPの解を構成して返す.

統合

再帰を用いたアルゴリズムの多くが分割統治も併用

注意事項

部分問題は，与えられた問題よりもサイズが小さくなければならない.

要書き込み

グリーディ法（greedy method）

- 欲張り法，貪欲法
- 基本方針
  - その時点で、局所的に最良のものを選ぶ
- 通常は，最適ではない解が求まる
  - 問題によっては、最適解が求まることもある
- 最適解が求まる例
  - 最短経路問題に対するダイクストラ法
  - 貨幣交換問題
  - 最小全域木（クラスカルのアルゴリズム）

硬貨の交換問題

- 問題
  - 50円玉,10円玉,5円玉, 1円玉がある。
  - N円をこれらの硬貨で支払うとき、枚数を最小にしたい。
- グリーディ法に基づくアルゴリズム
  1. Nを50で割り，商を50円玉の枚数とする.
  2. 残金＝ N-50×50円玉の枚数を求める.
  3. 残金を10で割り，商を10円玉の枚数とする.
  4. 残金から10×10円玉の枚数を引く.
  5. 残金を5で割り，商を5円玉の枚数とする.
  6. 残金から5×5円玉の枚数を引き，1円玉の枚数とする.

方針：大きい金額の硬貨に優先的に交換する

硬貨の交換問題

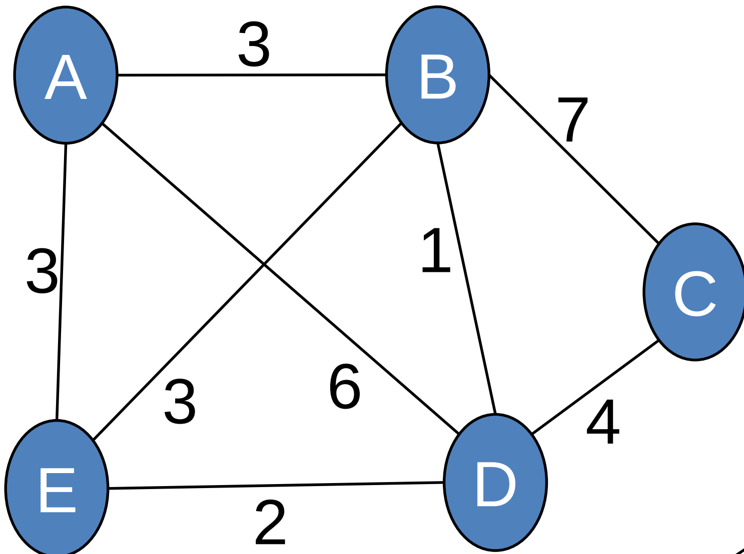
- グリーディ法でうまく行くことも多い
  - 硬貨の種類＝{ 50円, 10円, 5円, 1円}
  - N=127円
  - 最小枚数は合計7枚
    - 50円×2枚、10円×2枚, 5円×1枚, 1円×2枚
- 硬貨の種類によっては上手くいかない場合がある
  - 硬貨の種類＝{12ペンス, 5ペンス, 1ペニー}
  - N＝16ペンス
  - 最小枚数は合計4枚
    - 5ペンス×3枚, 1ペニー×1枚
  - グリーディ法では合計5枚
    - 12ペンス×1枚, 1ペニー×4枚

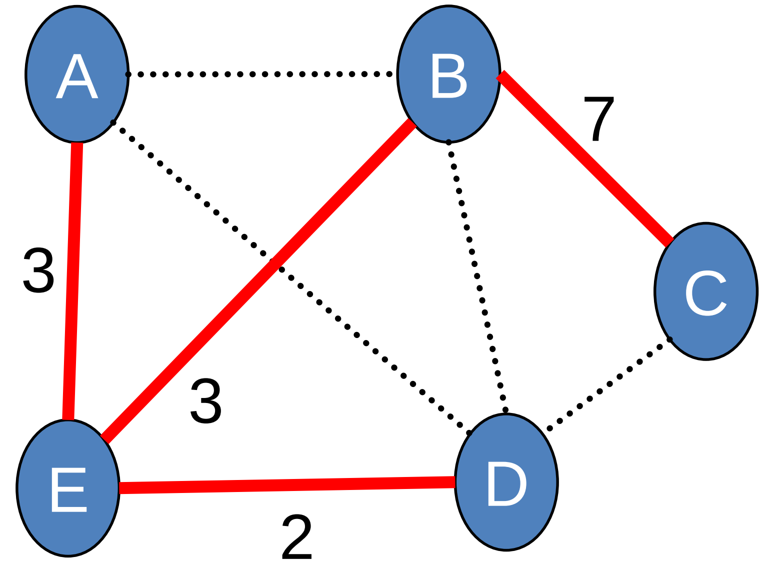
最小全域木問題

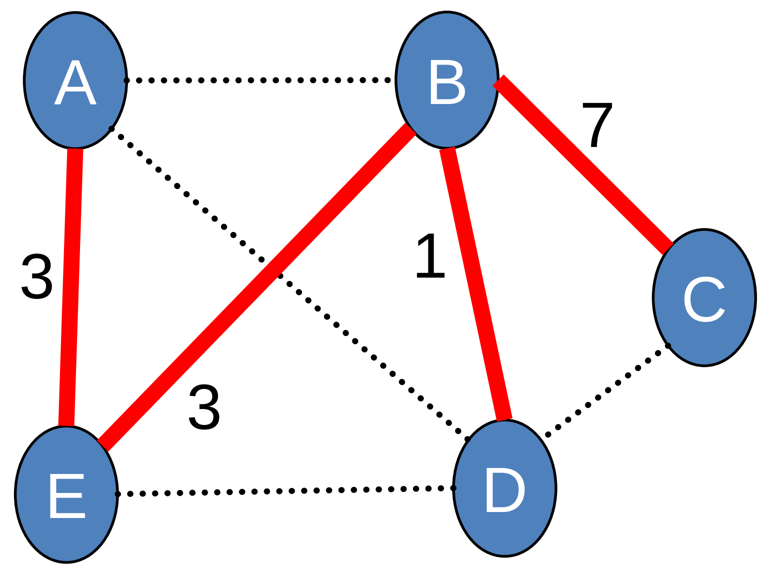
- 全域木（Spanning Tree）
  - 重み付き無向グラフG=（V, E）の部分木G'=（V, E'）
    - 頂点集合VはGの頂点集合と同一
    - 辺の集合E' ⊆ E は閉路を含まない
    - 全ての頂点が少なくとも1つの辺に含まれる
- 最小全域木（Minimum Spanning Tree, MST）問題
  - 入力：重み付き無向グラフG=（V,E）
  - 出力：Vの全域木のうち、辺の重みの和が最小のものを求めよ

応用事例：複数の村を互いに行き来できるように道路を作りたい．工事区間の長さが最短になるような計画を立てよ

全域木の例



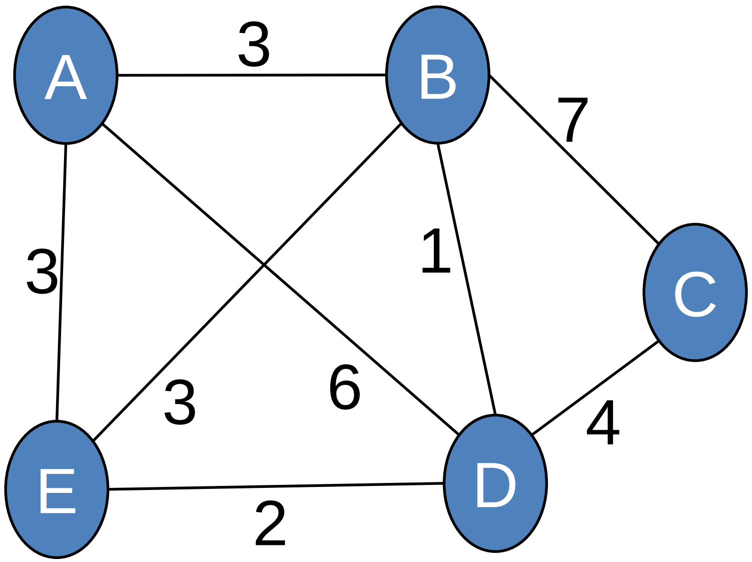


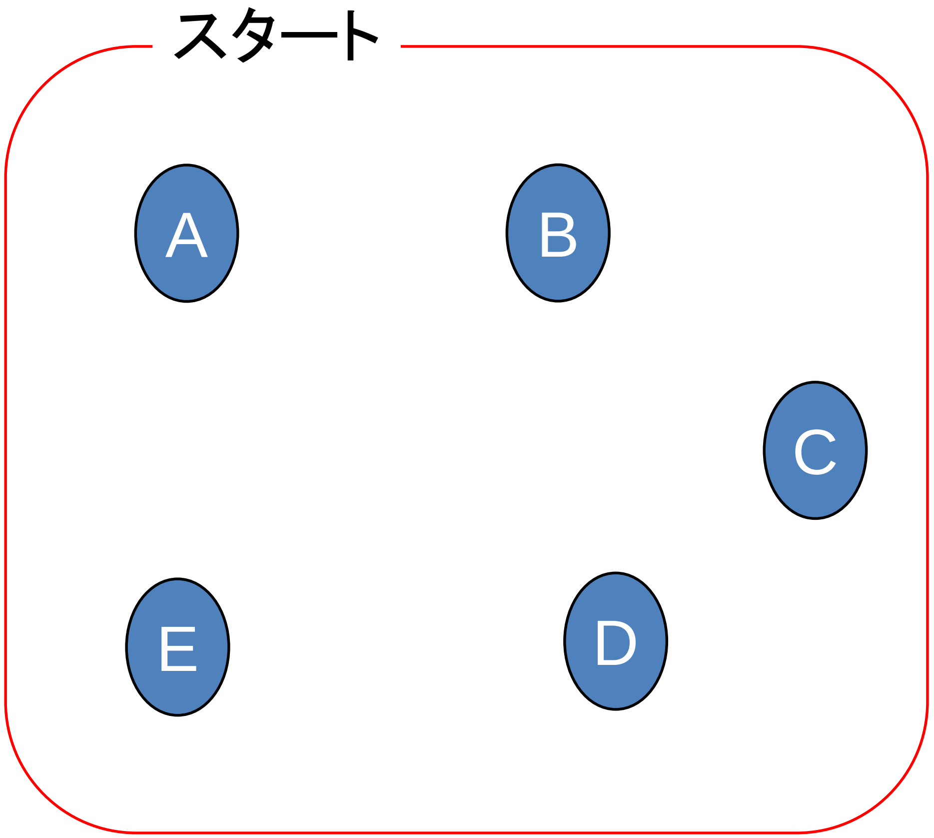


(重みの和)=15

(重みの和)=14

クラスカルのアルゴリズム(例)

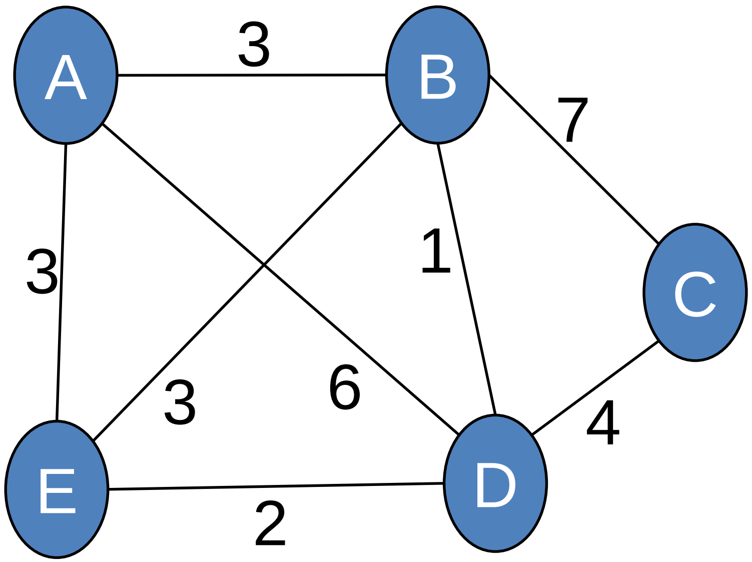


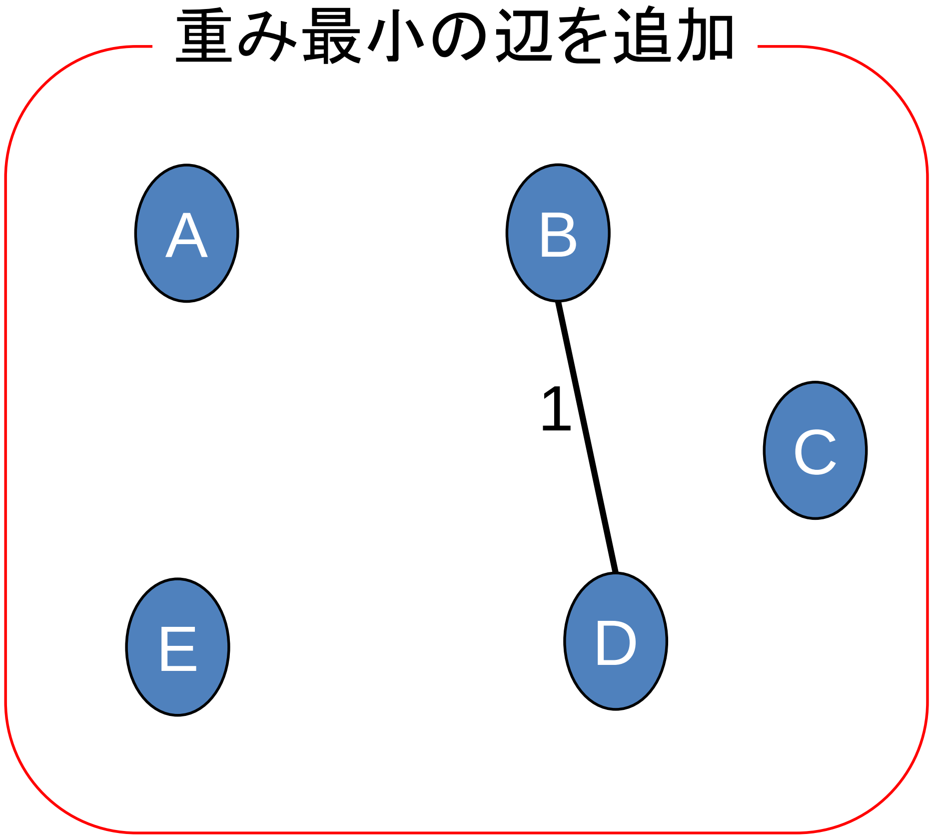


基本方針

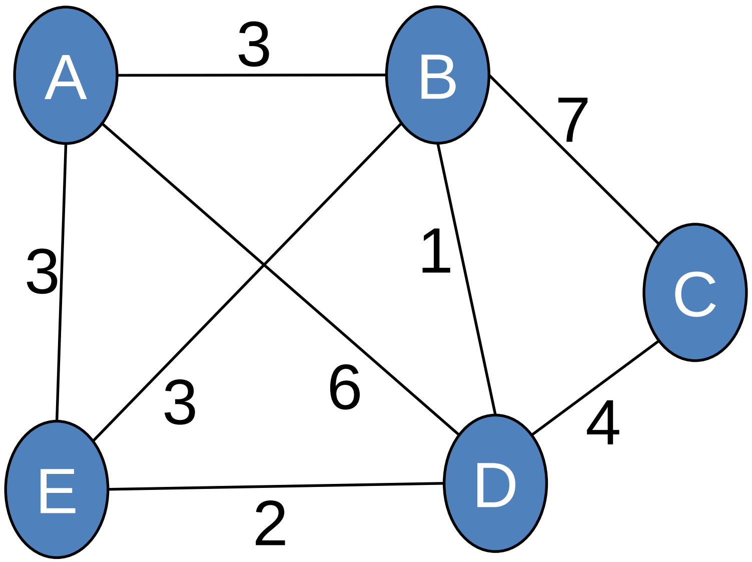
- 辺の重みが小さい順に選ぶ.
- ただし, 木でなくなる(巡回路ができる)場合には選ばない.

クラスカルのアルゴリズム（例）

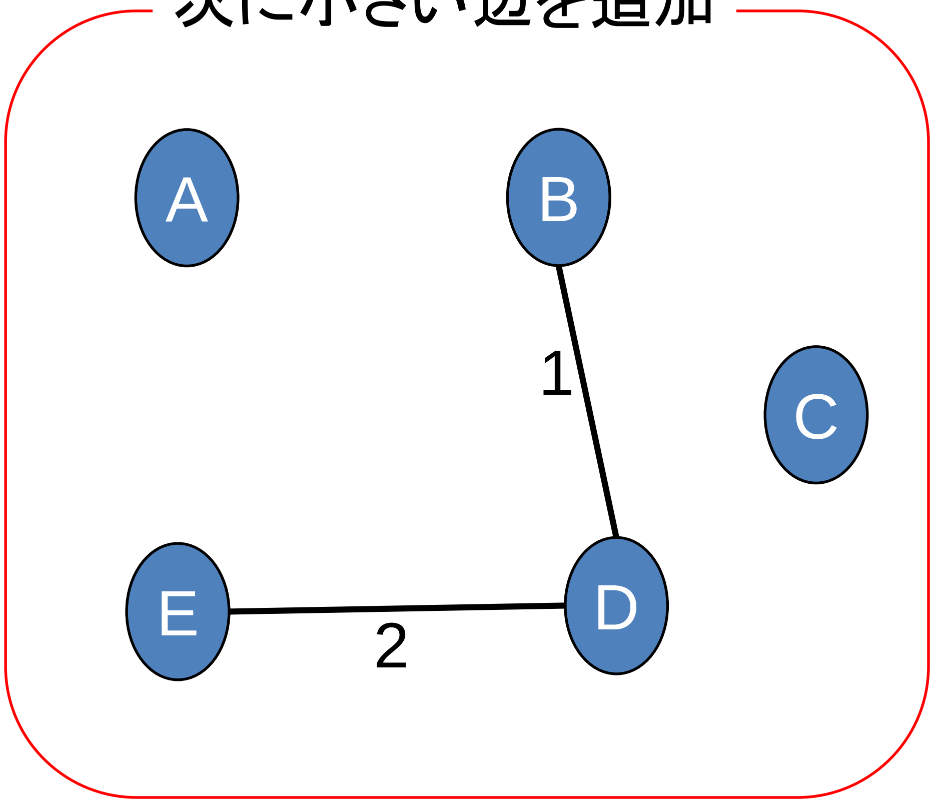




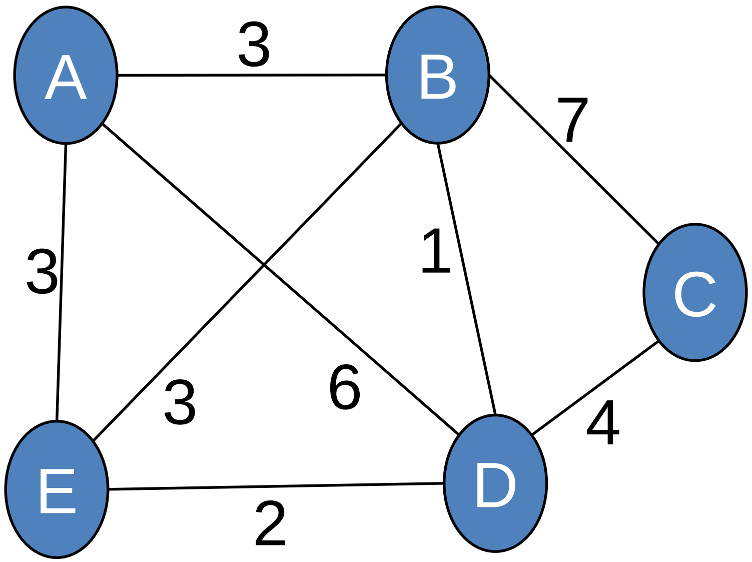
クラスカルのアルゴリズム(例)

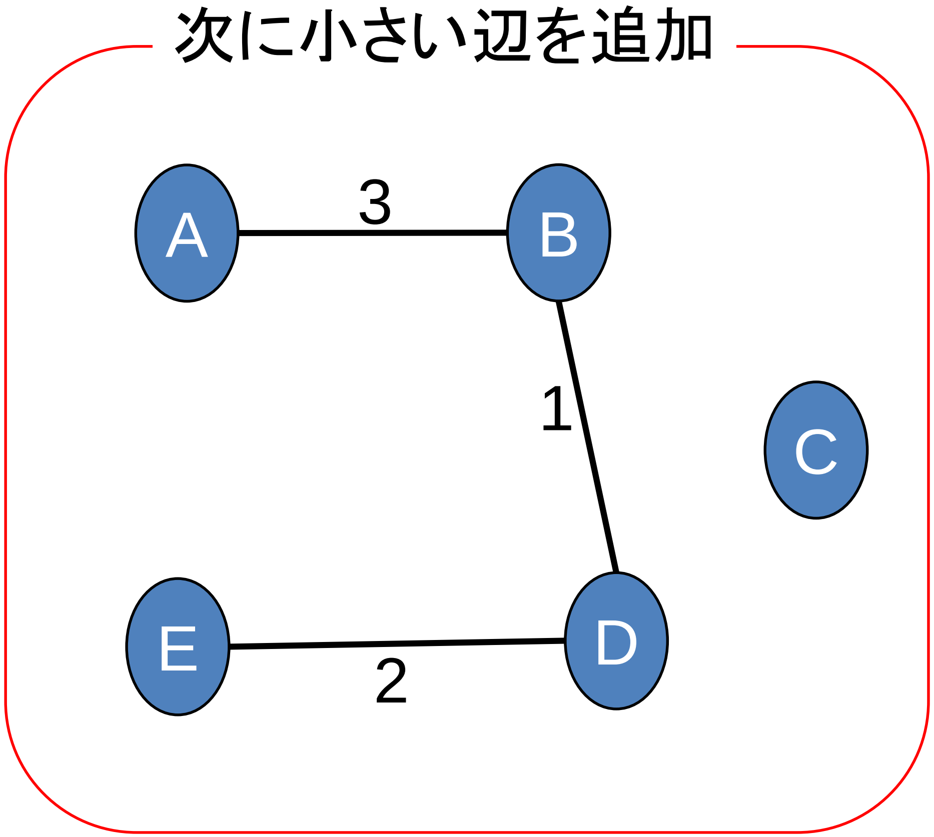


次に小さい辺を追加

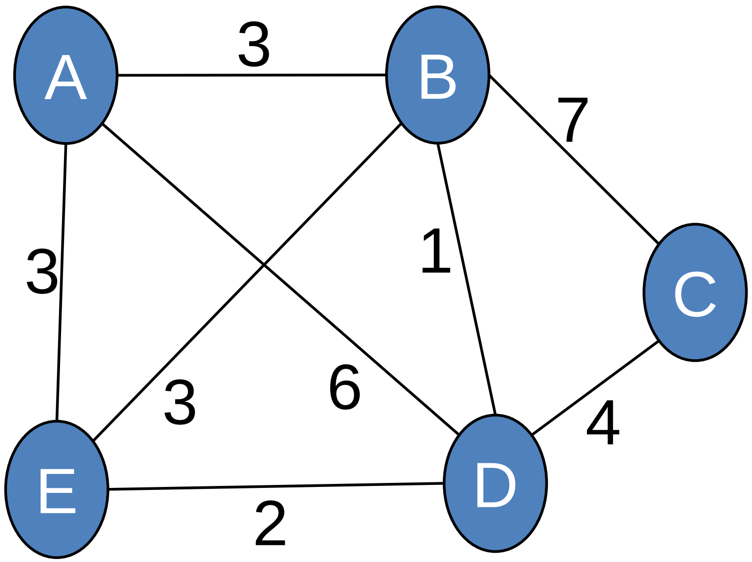


クラスカルのアルゴリズム(例)

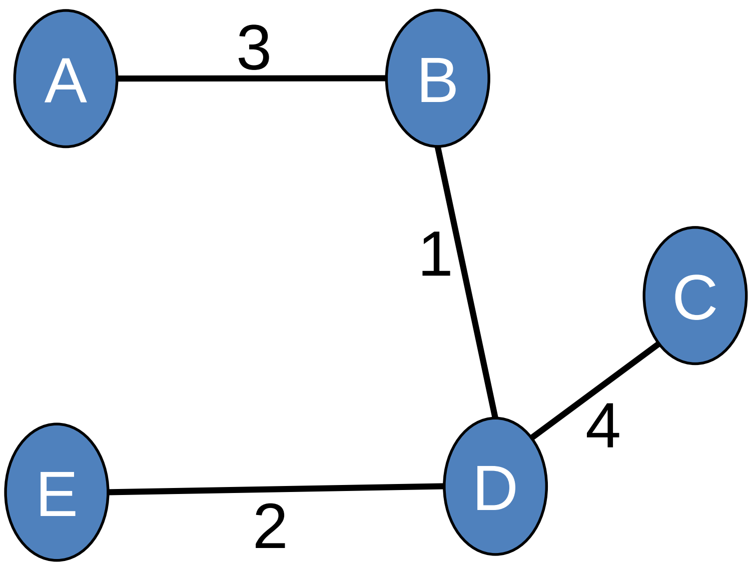




クラスカルのアルゴリズム(例)



次に小さい辺を追加
(※3の辺はこれ以上追加不能)



解(これが最適)

(重みの和)=10

定理

- クラスカルのアルゴリズムを用いると，任意の重み付きグラフについて最小全域木(MST)を求められる．

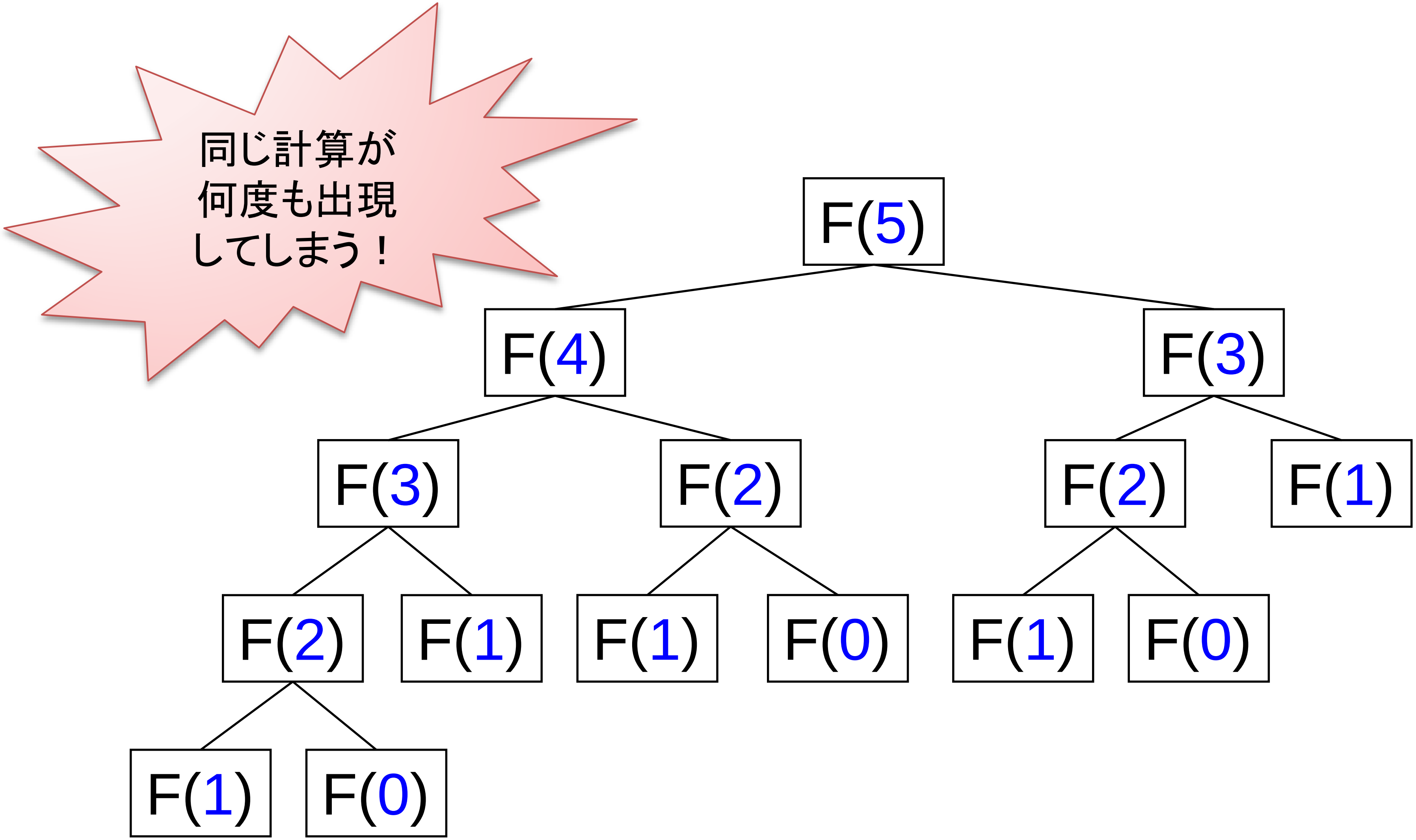
動的計画法(dynamic programming)

- 基本方針
  - サイズの小さな部分問題から順番に解き、結果を記録していく。
  - 目的サイズまで達したら終わり
- 特徴
  - 計算結果を記録することで計算量を削減
  - その代り，記憶領域が余分に必要になる場合が多い
- 代表例
  - フィボナッチ数の計算
  - 硬貨の交換問題
  - ナップサック問題

フィボナッチ数の計算

- フィボナッチ数
  - 定義
    - F(0) = F(1) = 1
    - F(n) = F(n-1) + F(n-2), n≧2の場合
- アルゴリズム（再帰版）
  1. n=0 または n=1 ならば 1 を返す.
  2. そうでなければ, F(n-1)+F(n-2)を返す.

フィボナッチ数を再帰で解くと．．



フィボナッチ数の計算（動的計画法）

添字	0	1	2	3	4	5
F	1	1	2	3	5	8

ステップ1
F(0) = 1

ステップ2
F(1) = 1

ステップ3
F(2) = F(1)+F(0)

ステップ4
F(3) = F(2)+F(1)

ステップ5
F(4) = F(3)+F(2)

ステップ6
F(5) = F(4)+F(3)

- 同じ値は1度しか計算しない． ⇒ 高速
- F(0)～F(5) を保持する領域が必要

要書き込み

フィボナッチ数の計算(改良版)

		A	B
ステップ1	F(0) = 1	1	
ステップ2	F(1) = 1		1
ステップ3	F(2) = F(1)+F(0)	2	
ステップ4	F(3) = F(2)+F(1)		3
ステップ5	F(4) = F(3)+F(2)	5	
ステップ6	F(5) = F(4)+F(3)		8

動的計画法を適用しつつ，必要な変数を2個に削減

硬貨の交換問題

- 問題
  - 12ペンス，5ペンス，1ペニーの硬貨がある。
  - 16ペンスをこれらの硬貨で支払うとき、枚数を最小にしたい。
- 正解
  - 4枚（5ペンス×3枚＋1ペニー×1枚）
- 動的計画法を用いたアルゴリズムの方針
  1. 1ペニー硬貨のみで支払う場合の解を求める
  2. 1ペニー硬貨と5ペンス硬貨を組み合わせて支払う場合の解に拡張
  3. 3種類の硬貨を組み合わせて支払う場合の解に拡張

1ペニー硬貨のみで支払う場合の解を求める

支払金額	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
硬貨の枚数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

硬貨の交換問題（動的計画法）

1ペニー硬貨と5ペンス硬貨で支払う場合

支払金額	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
硬貨の枚数	0	1	2	3	4	1	2	3	4	5	2	3	4	5	6	3	4

1ペニー硬貨のみで
支払う場合と同一

比較対象（n≧5の場合）
場合1：n-1ペンスに1ペニー硬貨を
加えてnペンスを支払う.
場合2：n-5ペンスに5ペンス硬貨を
加えてnペンスを支払う.

支払金額	0	1	2	3	4
硬貨の枚数	0	1	2	3	4

5ペンスをどう支払うか？

場合1：4ペンスに1ペニー硬貨を加えて支払う → 硬貨の枚数 = 4+1 = 5

場合2：0ペンスに5ペンス硬貨を加えて支払う → 硬貨の枚数 = 0+1 = 1

少ない方を採用

要書き込み

硬貨の交換問題(動的計画法)

3種類の硬貨を組み合わせて支払う場合

支払金額	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
硬貨の枚数	0	1	2	3	4	1	2	3	4	5	2	3					

1ペニー硬貨と5ペンス硬貨を組み合わせて支払う場合と同一

比較対象(n≧12の場合)

場合1:n-1ペンスに1ペニー硬貨を加えてnペンスを支払う.

場合2:n-5ペンスに5ペンス硬貨を加えてnペンスを支払う.

場合3:n-12ペンスに12ペンス硬貨を加えてnペンスを支払う.

3通りの場合について硬貨の必要枚数を求め，最も少ない場合を採用

動的計画法を用いると，硬貨の種類によらず，最適解を求められる

分枝限定法(branch and bound)

- 解候補を総当たりで探索する場合に有効
  - しらみつぶしに探索すると、場合の数が大きくなり過ぎる(例:将棋，チェスでの先読みなど)
- 見込みのない枝の探索は早めに打ち切る
- 「見込み」の有無は、問題によって判断基準が違う.
  - どのように判断するかが，工夫のしどころ

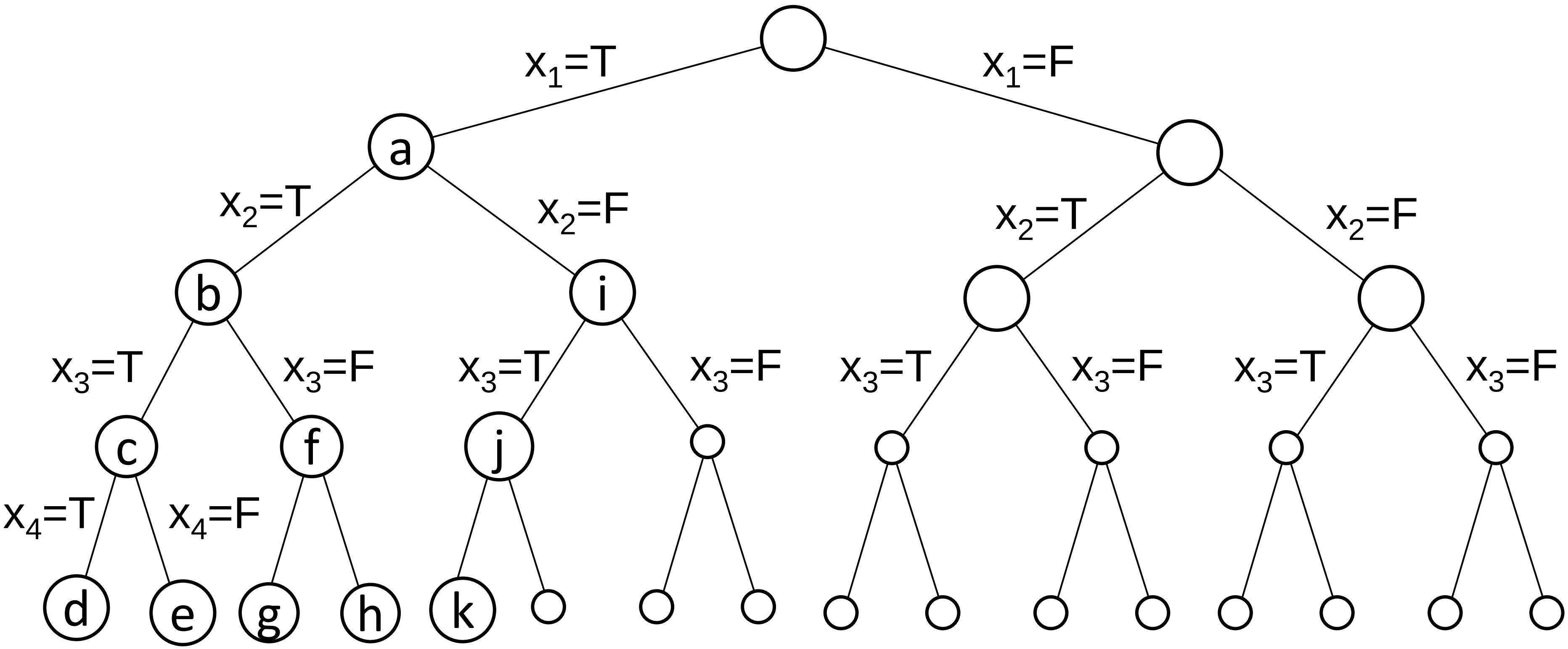
MAX-SAT問題

- 入力 : 論理式F
  - 例 : $F(x_1, x_2, x_3, x_4) = (x_1+\neg x_3)\cdot(x_1+x_2+\neg x_4)\cdot(\neg x_1+x_3)\cdot(x_1+\neg x_2+x_4)\cdot(x_2+x_3+\neg x_4)\cdot(x_1+\neg x_2+\neg x_4)\cdot(x_2)\cdot(x_1+x_4)\cdot(\neg x_1+\neg x_2)\cdot(x_1)$
- 出力 :
  - Fを構成する節について，充足する節の数が最も多くなるような変数$x_1, \ldots, x_n$の組み合わせを求めよ．

MAX-SAT問題の具体例

x_1	x_2	x_3	x_4	$x_1 + \neg x_3$	$x_1 + x_2 + \neg x_4$	$\neg x_1 + x_3$	$x_1 + \neg x_2 + x_4$	$x_2 + x_3 + \neg x_4$	$x_1 + \neg x_2 + \neg x_4$	x_2	$x_1 + x_4$	$\neg x_1 + \neg x_2$	x_1	項数
T	T	T	T	T	T	T	T	T	T	T	T	F	T	9
T	T	T	F	T	T	T	T	T	T	T	T	F	T	9
T	T	F	T	T	T	F	T	T	T	T	T	F	T	8
T	T	F	F	T	T	F	T	T	T	T	T	F	T	8
T	F	T	T	T	T	T	T	T	T	F	T	T	T	9
T	F	T	F	T	T	T	T	T	T	F	T	T	T	9
T	F	F	T	T	T	F	T	F	T	F	T	T	T	7
T	F	F	F	T	T	F	T	T	T	F	T	T	T	8
F	T	T	T	F	T	T	T	T	F	T	T	F	F	6
F	T	T	F	F	T	T	T	T	T	T	F	F	F	6
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:

変数への割り当てを表す木



深さ優先探索を用いて変数への全ての割り当てを調べる

分枝限定法を用いたMAX-SATの解法

ノード	変数への割り当て	充足される節	充足されない節	未定節
a	x_1=T	x_1を含む節（6個）	0個	4個
b	x_1=x_2=T	x_1またはx_2を含む節（8個）	1個	1個
c	x_1=x_2=x_3=T	x_1, x_2またはx_3を含む節（9個）	1個	0個
d	ノードcにて充足できる節数が確定　⇒　探索不要			
e	ノードcにて充足できる節数が確定　⇒　探索不要			
f	x_1=x_2=T, x_3=F	x_1, x_2または¬x_3を含む節（7個）	2個	1個
g	ノードfにて充足できる節数が8以下になることが確定　⇒　探索不要			
h	ノードfにて充足できる節数が8以下になることが確定　⇒　探索不要			
i	x_1=T, x_2=F	x_1,または¬x_2を含む節（7個）	1個	2個
j	x_1=x_3=T, x_2=F	x_1, ¬x_2またはx_3を含む節（7個）	2個	1個
k	ノードjにて充足できる節数が8以下になることが確定　⇒　探索不要			

（以下略）

まとめ：アルゴリズムの設計手法

様々なアルゴリズムを設計する際に使われる共通の戦略

再帰（recursion）

分割統治法（divide and conquer）

- 問題を分割し，個別に解く．それらの解を利用して全体の解を得る．
- 再帰と組み合わせて使う場合も多い（例：クイックソート，マージソートなど）．

グリーディ法（greedy method，欲張り法，貪欲法）

- 各時点で局所的に最善のものを選んでいく．

動的計画法（dynamic programming）

- サイズの小さな問題から順番に解きながら解を記録．
- 記録した解を活用して全体の解を得ると同時に処理を高速化

分枝限定法（branch and bound）

- 解候補を総当たりで探すが，見込みの無い探索は早めに打ち切る．

確認テスト(第14週)

- ハノイの塔
  - 再帰アルゴリズムの手間
- フィボナッチ数の計算
  - 再帰アルゴリズムの手間
  - 動的計画法アルゴリズムの手間
- 硬貨の交換問題(動的計画法)
  - 最適解を求めるための式の定義