アルゴリズムとデータ構造III
7回目：11月19日（金）4時限

グラフ
（動的計画法，ダイクストラ法， A*アルゴリズム，DPマッチング）

授業資料
http://ir.cs.yamanashi.ac.jp/~ysuzuki/public/algorithm3/

授業の予定（中間試験まで）

1	10/07	スタック（後置記法で書かれた式の計算）
2	10/14	チューリング機械，文脈自由文法
3	10/21	構文解析　CYK法
4	11/04	構文解析　CYK法
5	11/11	構文解析（チャート法）
6	11/18	構文解析（チャート法），グラフ（ダイクストラ法，DPマッチング）
7	11/19 4時限 B2-41	グラフ（ A*アルゴリズム，DPマッチング）
8	11/25	グラフ（DPマッチング），前半のまとめ
9	12/02	中間試験

授業の予定（中間試験以降）

10	12/09	全文検索アルゴリズム（simple search, KMP)
11	12/16	全文検索アルゴリズム（BM, Aho-Corasick)
12	01/06	全文検索アルゴリズム（Aho-Corasick)，データ圧縮
13	01/13	暗号（黄金虫，踊る人形） 符号化（モールス信号，Zipfの法則，ハフマン符号）テキスト圧縮
14	01/20	テキスト圧縮　（zip)， 音声圧縮　（ADPCM，MP3，CELP)， 画像圧縮（JPEG）
15	02/03	**期末試験**

本日のメニュー

- 動的計画法
- ダイクストラ法（復習）
  - 動作例
  - アルゴリズム
- A*アルゴリズム
  - 動作例
  - アルゴリズム
- DPマッチング

動的計画法(Dynamic Programming)

- 部分問題の解をより大きな問題を解くために利用
- 同じ問題を2度解かなくても済むように解を格納
- 最適解に利用できない部分問題は省略する

- アルゴリズムの例
  - CYK法（構文解析）
  - ダイクストラ法（最短経路問題）
  - DPマッチング（パターンマッチング　DNAの解析にも利用）
  - DPを使った解法（ナップサック問題）
  - ビタビアルゴリズム（音声認識など）

ダイクストラ法

- 動的計画法を最短経路問題に適用

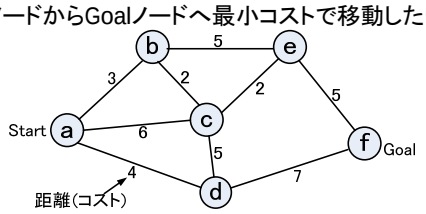
- 最適経路中の部分経路もまた最適経路になっている

身近な最短経路問題

- 道路の経路探索（カーナビなど）

ダイクストラ法（最短経路問題用アルゴリズム）

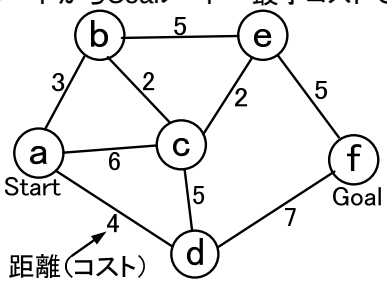
- StartノードからGoalノードへ最小コストで移動したい



- a-e, a-dなどの最短距離をa-fの最短距離を見つけるために利用
- 部分問題の解をより大きな問題を解くために利用
- 同じ問題を2度解かなくても済むように解を格納

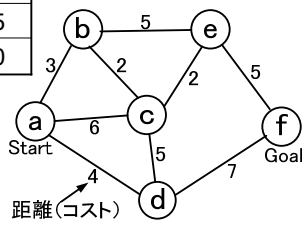
ダイクストラ法（最短経路問題用アルゴリズム）

- StartノードからGoalノードへ最小コストで移動したい



隣接行列（コスト付き）

	a	b	c	d	e	f
a	0	3	6	4	∞	∞
b	3	0	2	∞	5	∞
c	6	2	0	5	2	∞
d	4	-	5	0	∞	7
e	∞	5	2	∞	0	5
f	∞	∞	∞	7	5	0

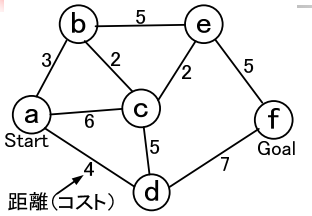


ダイクストラ法　アルゴリズム

1. 初期化：
   1. コスト表：各ノードとスタートノード間のコストをコスト表に書き出す（スタートノードと隣接していないノードのコストは∞，スタートノードのコストは0とする）
   2. 親ノード表：各ノードの直前のノードとしてスタートノードを親ノード表に書き出す
2. 未確定ノードが無くなるまで以下の処理を繰り返す．
   1. 未確定ノードのうち，最小コストのノードを見つけ確定ノードとする
   2. 新しい確定ノードから隣接する未確定ノードを探す
   3. 隣接する未確定ノードに対して「確定ノードまでのコスト＋確定ノードから未確定ノードまでのコスト」を計算する．計算結果がそのノードのコスト表の値よりも小さければ
      1. コスト表：計算結果を未確定ノードまでのコストとする
      2. 親ノード表：新しい確定ノードを未確定ノードの親ノードとする

ダイクストラ法　動作例　1/20

コスト行列の作成

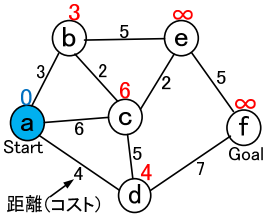


	a	b	c	d	e	f
a	0	3	6	4	∞	∞
b	3	0	2	∞	5	∞
c	6	2	0	5	2	∞
d	4	∞	5	0	∞	7
e	∞	5	2	∞	0	5
f	∞	∞	∞	7	5	0

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　2/20

各ノードのStartノードからのコスト，親ノードを調べる
aのコストと親ノードは確定

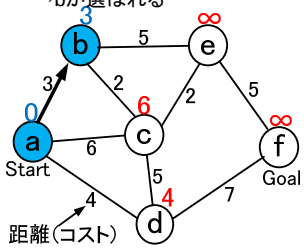


ノード	コスト	親ノード
a	0	a
b	3	a
c	6	a
d	4	a
e	∞	a
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　3/20

未確定ノードのうち，最小コストのノードを選び確定リストとする
→bが選ばれる

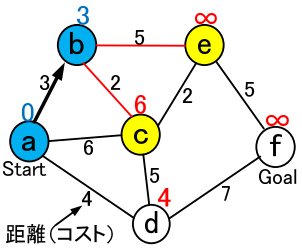


ノード	コスト	親ノード
a	0	a
b	3	a
c	6	a
d	4	a
e	∞	a
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　4/20

新たに確定ノードになったbに隣接する未確定ノードを探す:c,e

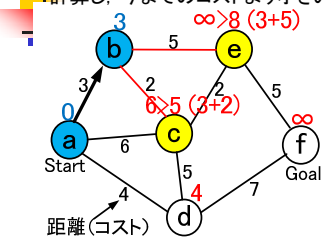


ノード	コスト	親ノード
a	0	a
b	3	a
c	6	a
d	4	a
e	∞	a
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　5/20

隣接する未確定ノードごとに「bまでのコスト＋bからのコスト」を
計算し，今までのコストより小さい場合はコストを書き換える

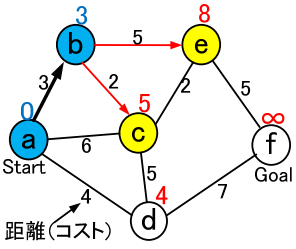


ノード	コスト	親ノード
a	0	a
b	3	a
c	6>5	a→b
d	4	a
e	∞>8	a→b
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　6/20

eの親ノードとcの親ノードをbに書き換える

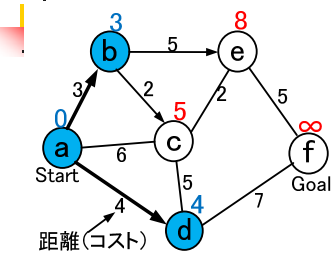


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　7/20

未確定ノードのうち，最小コストのノードを選ぶ→dが選ばれる

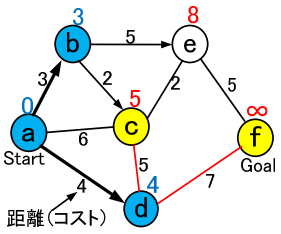


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　8/20

新たに確定ノードになったdに隣接する未確定ノードを探す:c,f

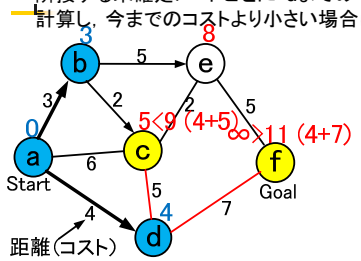


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	∞	a

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　9/20

隣接する未確定ノードごとに「dまでのコスト＋dからのコスト」を
計算し，今までのコストより小さい場合はコストを書き換える

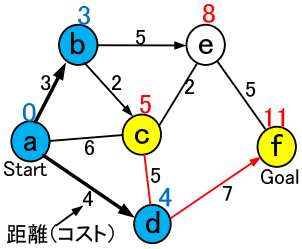


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	∞>11	a→d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　10/20

.fの親ノードをdに書き換える

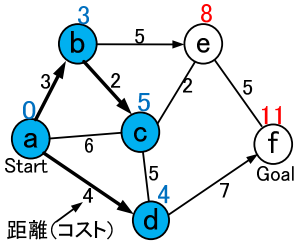


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　11/20

未確定ノードのうち，最小コストのノードを選ぶ→cが選ばれる

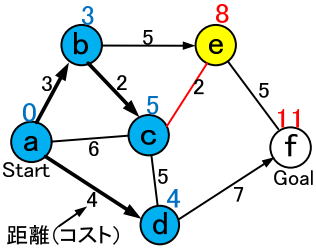


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　12/20

新たに確定ノードになったcに隣接する未確定ノードを探す:e

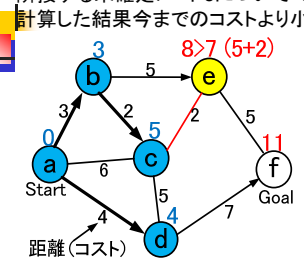


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8	b
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　13/20

隣接する未確定ノードeについて「cまでのコスト＋cからのコスト」を
計算した結果今までのコストより小さくなったのでコストを書き換え

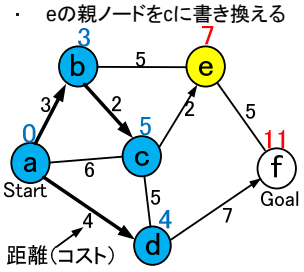


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	8>7	b→c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク
最短経路再計算用に使用
StartからGoalまでの
最短経路

ダイクストラ法　動作例　14/20

- eの親ノードをcに書き換える

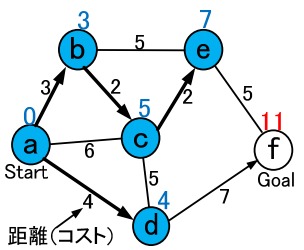


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　15/20

- 未確定ノードのうち，最小コストのノードを選ぶ→eが選ばれる

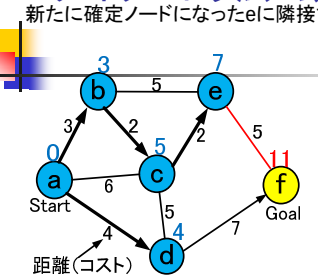


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　16/20

- 新たに確定ノードになったeに隣接する未確定ノードを探す:f

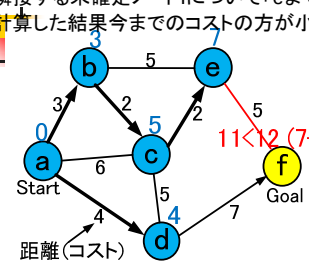


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　17/20

- 隣接する未確定ノードfについて「eまでのコスト＋eからのコスト」を計算した結果今までのコストの方が小さいのでコストは書き換えない

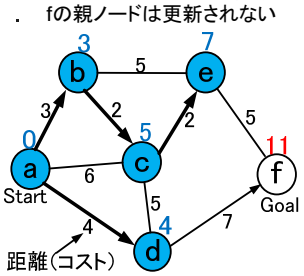


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11<12	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　18/20

- fの親ノードは更新されない

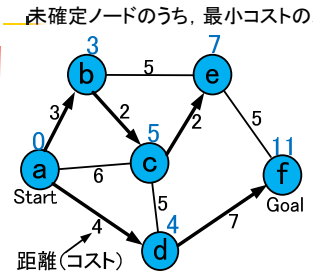


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　19/20

- 未確定ノードのうち，最小コストのノードを選ぶ→fが選ばれる

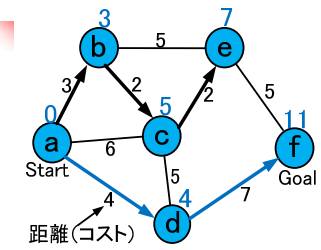


ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

Startからの最短経路が確定していないノード
Startからの最短経路再計算中のノード
Startからの最短経路が確定したノード
7 Startからの最短距離候補（未確定）
7 Startからの最短距離（確定済）
確定済ノードからのアーク 最短経路再計算用に使用
StartからGoalまでの最短経路

ダイクストラ法　動作例　20/20

新たに確定ノードになったfに隣接する未確定ノードを探す:無し → 終了



ノード	コスト	親ノード
a	0	a
b	3	a
c	5	b
d	4	a
e	7	c
f	11	d

- Startからの最短経路が確定していないノード
- Startからの最短経路再計算中のノード
- Startからの最短経路が確定したノード
- 7 Startからの最短距離候補(未確定)
- 7 Startからの最短距離(確定済)
- 確定済ノードからのアーク 最短経路再計算用に使用
- StartからGoalまでの最短経路

ダイクストラ法　アルゴリズム

1. 初期化:
   1. コスト表:各ノードとスタートノード間のコストをコスト表に書き出す(スタートノードと隣接していないノードのコストは∞，スタートノードのコストは0とする)
   2. 親ノード表:各ノードの直前のノードとしてスタートノードを親ノード表に書き出す
2. 未確定ノードが無くなるまで以下の処理を繰り返す.
   1. 未確定ノードのうち，最小コストのノードを見つけ確定ノードとする
   2. 新しい確定ノードから隣接する未確定ノードを探す
   3. 隣接する未確定ノードに対して「確定ノードまでのコスト＋確定ノードから未確定ノードまでのコスト」を計算する．計算結果がそのノードのコスト表の値よりも小さければ
      1. コスト表:計算結果を未確定ノードまでのコストとする
      2. 親ノード表:新しい確定ノードを未確定ノードの親ノードとする

ダイクストラ法の特徴

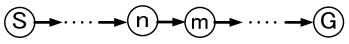
- 最短経路の見つけ方
  - ゴールノードから「どこから来たのか」調べ，さかのぼる(距離更新時に直前のノードを記述しておく).
- マイナスのコストを持つエッジは扱えない.
- 特定のノードからの最短距離およびその経路が全てのノードに対して求まる.
- 情報処理技術者試験で頻出
  - 平成15年秋期　基本情報技術者試験　午後の問題
  - 応用情報技術者試験にも出題

最短経路問題(より効率の良いアルゴリズム)

- ダイクストラ法は各ノードとスタートの間のコストだけに注目
- もし各ノードとゴールの間のコストが予想できれば効率よく探索可能

A*アルゴリズム

- 評価関数 $\hat{f}(m)$ が小さい順に経路を探索



$\hat{g}(n)$　$w(n,m)$　$\hat{h}(m)$

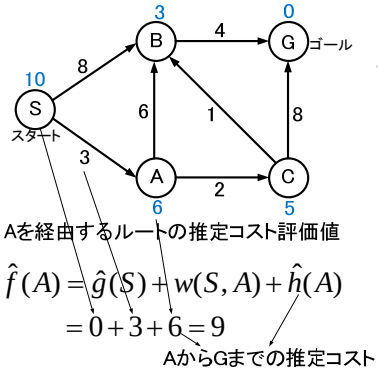
$$\hat{f}(m) = \hat{g}(n) + w(n,m) + \hat{h}(m)$$

ただし $0 \le \hat{h}(m) \le h(m)$

A*アルゴリズム
最短経路探索問題

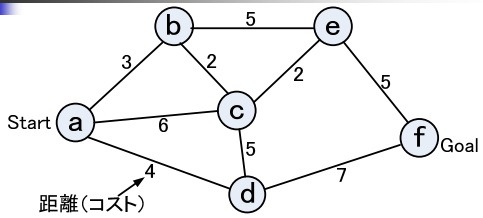
- ダイクストラ法にすこし工夫を加えた方法
- 各ノードからゴールまでの推定距離を利用
  - 0≦推定距離≦最短距離でなければならない
  - 推定距離=0なら推定していないと同じ→ダイクストラ法

まずはAアルゴリズム

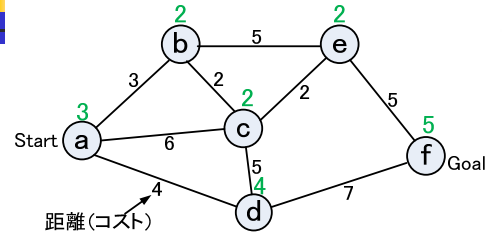


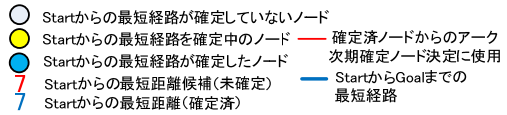
$\forall n, 0 \le \hat{h}(n) \le h(n)$
を満たすとき
A^*アルゴリズム
但し，nは節点番号，
$h(n)$は節点nから
ゴールまでのコスト

A*アルゴリズム　動作例

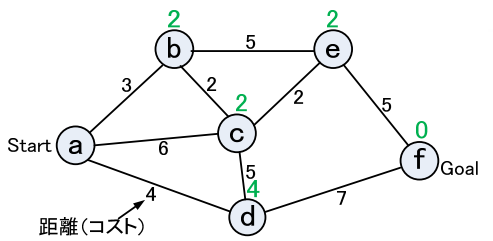


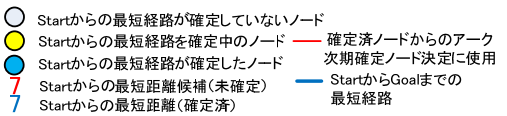
A*アルゴリズム　動作例 1/14



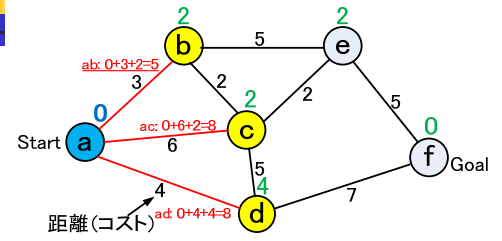


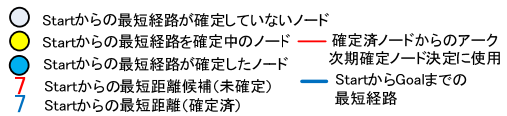
A*アルゴリズム　動作例 2/14



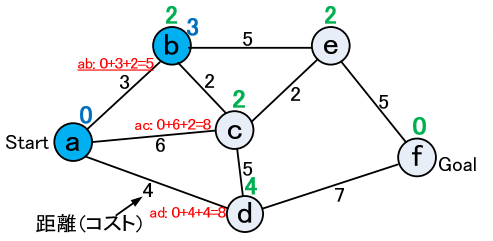


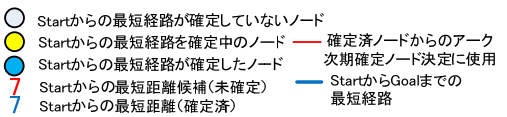
A*アルゴリズム　動作例 3/14



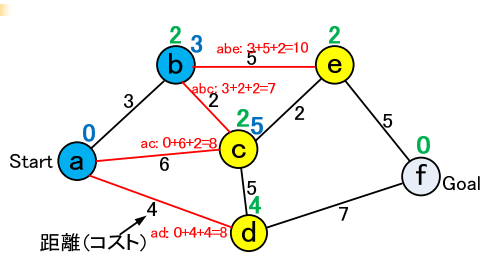


A*アルゴリズム　動作例 4/14



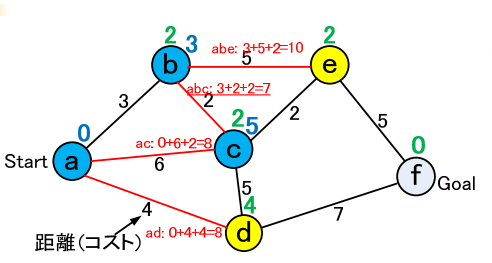


A*アルゴリズム　動作例 5/14



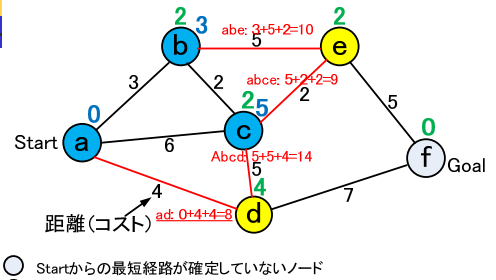
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 6/14



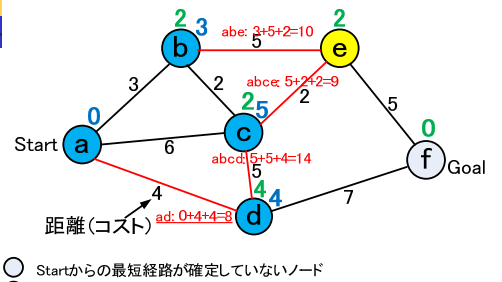
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 7/14



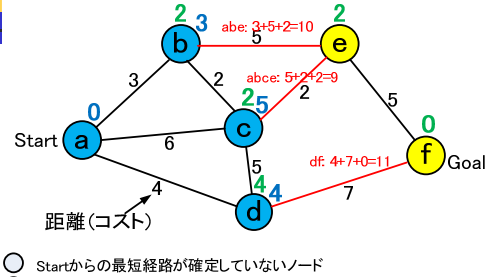
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 8/14



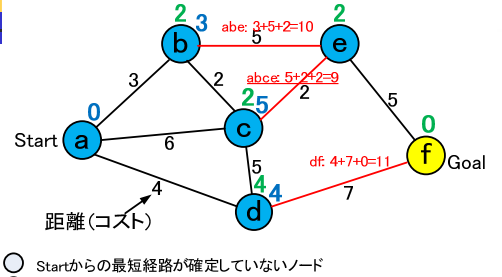
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 9/14



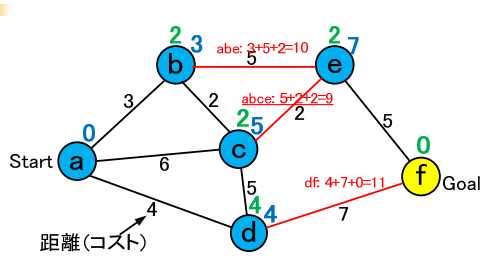
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 10/14



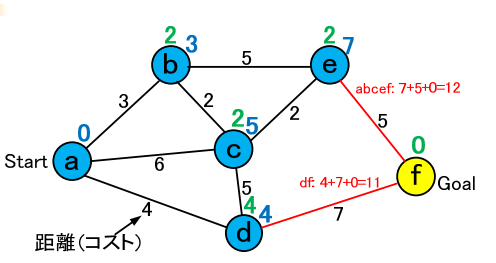
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 11/14



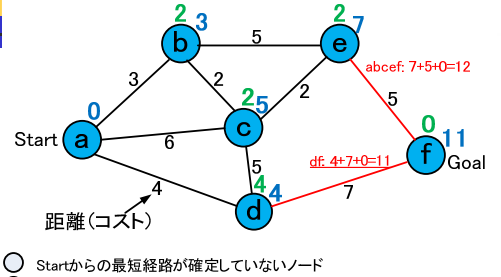
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 12/14



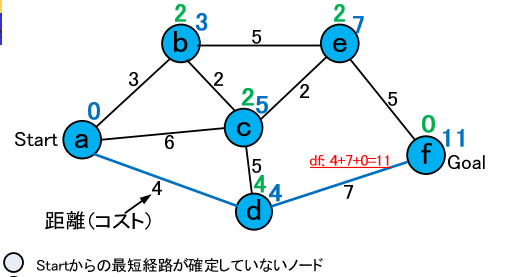
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 13/14



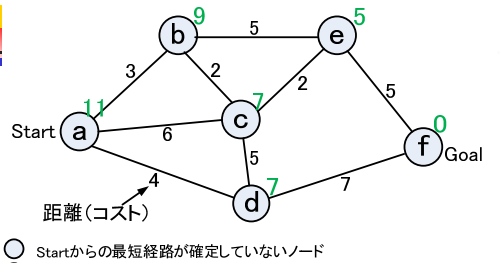
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 14/14



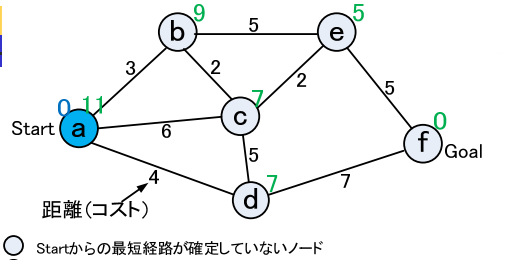
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム (最良の$\hat{h}(m)$)　1/6



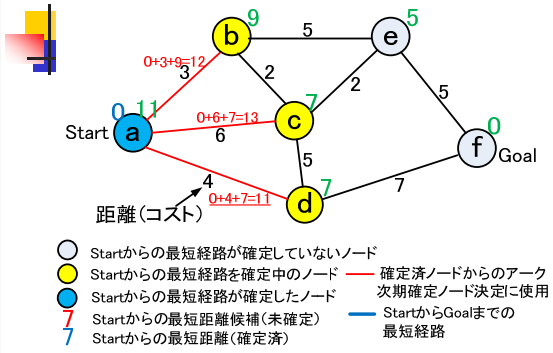
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　その2　2/6

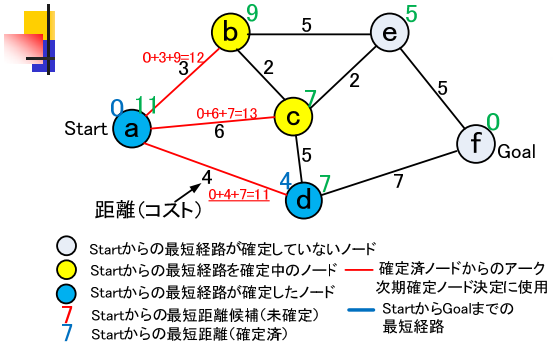


Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補(未確定)
Startからの最短距離(確定済)
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

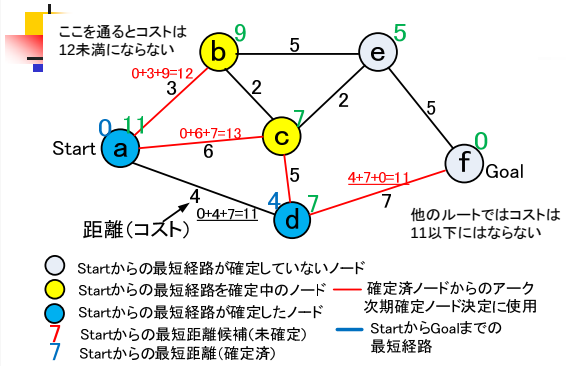
A*アルゴリズム その2 3/6



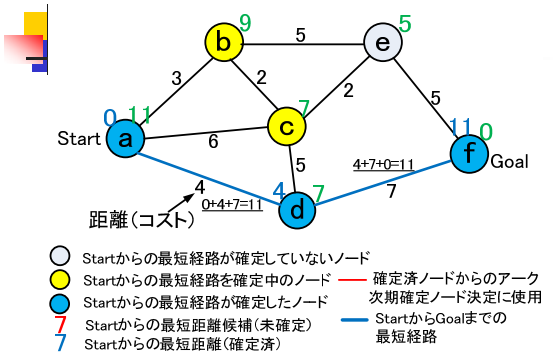
A*アルゴリズム その2 4/6



A*アルゴリズム その2 5/6



A*アルゴリズム その2 6/6



Aアルゴリズム，A*アルゴリズム ダイクストラ法

- f(n)=g(n)+h(n)
  - n:節点番号，
  - f(n):節点nを通りスタートからゴールまでの距離
  - g(n):スタートから節点nまでの距離
  - h(n):節点nからゴールまでの距離
- Aアルゴリズム
  - f*(n)=g(n)+h*(n)を利用してスタートからゴールまでの距離を調べる
    - f*(n): 節点nからゴールまでの距離の評価値
    - h*(n): 節点nからゴールまでの距離の評価値
- A*アルゴリズム
  - Aアルゴリズムのf*(n)=g(n)+h*(n)のh*(n)が$0 \leqq h^*(n) \leqq h(n)$の時
  - 最初に見つけたルートが最短ルートであることが保証されている
- ダイクストラ法
  - Aアルゴリズムのf*(n)=g(n)+h*(n)のh*(n)が0の時
  - つまりf*(n)=g(n)

DPマッチング （例：文字列の照合）

- 2つの文字列がどのくらい似ているかを調べる．
  - Yamanashi と kamonohashi, takahashi との比較
- 音声認識にも使える
  - 音声を文字列に変換した後，登録単語と比較
  - （現在主流の）HMM(Hidden Markov Model)に拡張可能
- DNAの比較にも使える
  - A（アデニン），G（グアニン），C（シトシン），T（チミン）の並び方の比較
  - ACTGAGCATTとCTGGACTACGの比較

ここまで

DPマッチング（例：文字列の照合）

- 簡単に比較できる例
  - abcdef
  - abzdef
- A: abcdef に対して脱落，挿入，置換
  - A: abcdef
  - B: abdef （脱落）
  - C: abccdef （挿入）
  - D: abzdef （置換）

DPマッチング：脱落，挿入，置換誤りを考慮して文字列照合可能