アルゴリズムとデータ構造III
8回目：11月20日

グラフ
（動的計画法，DPマッチング，A＊アルゴリズム）

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

中間試験

- 中間試験日
  - 11月27日（木）
- 範囲
  - スタック
  - 文脈自由文法
  - 構文解析
  - CYK法
  - （トップダウンチャート法）
  - 動的計画法
  - ダイクストラ法
  - DPマッチング
  - A*アルゴリズム

アンケート結果1/3（良いところ）

- スライドを用いた分かりやすい説明
- パワポが見やすく分かりやすい
- 資料が分かりやすい．
- 説明がすごく丁寧で良い
- 説明が丁寧
- 先生の説明がとても丁寧
- 分かりやすいスライドと説明
- スライドが見やすい
- スライドを使っているところ
- スライド（3件）
- スライドが見やすくてとても分かりやすいです．後半もこの調子でお願いしたいです．
- スライドを用いて丁寧に説明してくれる
- 説明がゆっくりでわかりやすい
- 練習問題がある
- 練習問題と解答例があり，勉強がしやすい．理解が深まる
- 授業の途中に練習問題をやり，その解説をしてくれるところ
- 練習問題の答を解説してくれること
- 説明だけでなく例題を出すので分かりやすい
- 授業予定を出してくれるのでスケジュール（テスト勉強などの）がとりやすい
- 授業資料がホームページに載せられているので復習しやすい
- パワーポイントがWebにのっているので授業中に聞き逃しても復習が出来る
- パソコンで資料を印刷できる
- 講義のスライドを印刷できるところ
- 授業資料を見ることが出来るので復習しやすい
- スライドの資料があるので復習がしやすいところ
- スライドがWebにあること
- 授業で用いた授業資料が後で見られること
- 授業資料をあげてくれるので復習しやすい
- 授業のスライドがWebにのっていること
- アルゴリズムとデータ構造I，IIの復習もかねているところ
- 2限であること
- よく分かりません

アンケート結果2/3
改善して欲しいところ

- 練習問題がない場面が多かったので，1つでも良いので載せて欲しい
- 授業前にサイトに載せて欲しい
- よく分かりません

アンケート結果3/3
アンケートに対する対応

- 練習問題がない場面が多かったので，1つでも良いので載せて欲しい
  - →毎回練習問題を出すようにします．
- 授業前にサイトに載せて欲しい
  - →昨年の授業資料をWeb上に残してあるので，授業前は昨年度の資料を見てください．
  - →なるべく早くアップロードするようにします．
- よく分かりません
  - →分かりやすいように工夫します．

授業の予定（中間試験まで）

1	10/02	スタック（後置記法で書かれた式の計算）
2	10/09	チューリング機械，文脈自由文法
3	10/16	構文解析　CYK法
4	10/23	構文解析　CYK法
5	10/30	構文解析（チャート法），グラフ（ダイクストラ法）
6	11/06	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	11/13	グラフ（DPマッチング，A*アルゴリズム）
8	11/20	グラフ（A＊アルゴリズム），前半のまとめ
9	11/27	中間試験

授業の予定（中間試験以降）

10	12/04	全文検索アルゴリズム（simple search, KMP）
11	12/11	全文検索アルゴリズム（BM, Aho-Corasick）
12	12/18	全文検索アルゴリズム（Aho-Corasick），データ圧縮
13	01/08	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	01/15	テキスト圧縮 （zip）， 音声圧縮 （ADPCM，MP3，CELP）， 画像圧縮（JPEG）
15	01/29	期末試験

本日のメニュー

- 動的計画法
- DPマッチング
  - アルゴリズム
  - 動作
- A*アルゴリズム
- 中間試験の範囲の説明

動的計画法 (Dynamic Programming)

- 解くのに時間のかかる問題を、複数の部分問題に分割することで効率的に解くアルゴリズム

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

ダイクストラ法　アルゴリズム

1. 初期化：スタートノードの値（最小コスト候補）を0，他のノードの値を無限大に設定
2. 未確定ノードが無くなるまで以下のループを繰り返す．
   1. 確定中ノードのうち，最小の値を持つノードを見つけ，確定ノードとする．
   2. 確定ノードからのエッジに対して「確定ノードまでのコスト＋エッジのコスト」を計算し，そのノードの現在値よりも小さければ更新．

ダイクストラ法のアルゴリズム

- begin
  - for each x ∈V do begin
    - cost[x]:=w[s,x];
    - parent[x]:='s';
  - end　　costとparentの初期化
  - U:=V-{s};　　U（未確定ノード）の初期化
  - while U ≠φ do
  - begin
    - U中のmで，cost[m]が最小となる頂点mを選ぶ；
    - U:=U-{m};　　集積コストが最も小さいノードmを選んで，cost[m]とparent[m]を確定
    - mから隣接する頂点の集合をD_mとする；
    - for each x ∈ D_m ∩ U do　　頂点mから隣接するノードすべての集合Dmを求める．Dmの要素で且つ未確定ノードである各xについてmを経由してxに至る最短経路のコストを計算し，現在のcost[x]と比較し，小さければ更新する
      - If cost[m]+w[m,x]<cost[x]
      - then begin
        - Cost[x]:=cost[m]+w[m,x];
        - Parent[x]:=m
      - end
  - end
- end

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．
- 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．

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．
  - takeda は nakadaiとどのくらい似ているか
  - 置換，脱落，挿入に対応
- 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

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に対して脱落，挿入，置換
  - A: abcdef
  - B: abdef
  - C: abccdef
  - D: abzdef

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

DPマッチング（例：文字列の照合）1/8

takeda と nakadai の照合

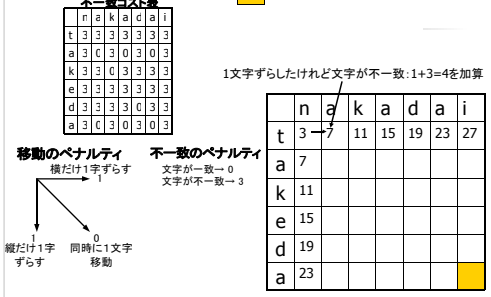
文字が一致→ 0
文字が不一致→ 3

不一致コスト表

	n	a	k	a	d	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

DPマッチング（例：文字列の照合）2/8

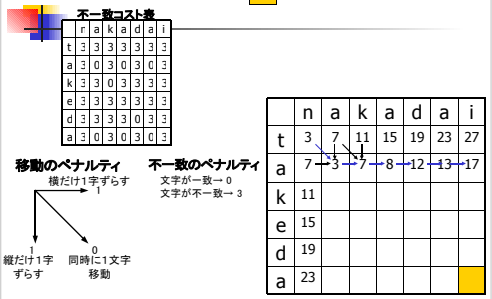
- takeda と nakadai



	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7						
k	11						
e	15						
d	19						
a	23						

DPマッチング（例：文字列の照合）3/8

- takeda と nakadai



	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11						
e	15						
d	19						
a	23						

DPマッチング（例：文字列の照合）4/8

- takeda と nakadai

の値を求める

不一致コスト表

	n	a	k	a	d	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11	7	3	7	11	15	16
e	15						
d	19						
a	23						

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

DPマッチング（例：文字列の照合）5/8

- takeda と nakadai

の値を求める

不一致コスト表

	r	a	k	a	d	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11	7	3	7	11	15	16
e	15	11	7	6	10	14	18
d	19						
a	23						

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

DPマッチング（例：文字列の照合）6/8

- takeda と nakadai

の値を求める

不一致コスト表

	n	a	k	a	d	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11	7	3	7	11	15	16
e	15	11	7	6	10	14	18
d	19	15	11	10	6	10	14
a	23						

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

DPマッチング（例：文字列の照合）7/8

- takeda と nakadai

の値を求める

不一致コスト表

	r	a	k	a	c	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11	7	3	7	11	15	16
e	15	11	7	6	10	14	18
d	19	15	11	10	6	10	14
a	23	16	15	11	10	6	10

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

DPマッチング（例：文字列の照合）8/8

- takeda と nakadai

の値を求める

不一致コスト表

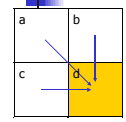
	n	a	k	a	d	a	i
t	3	3	3	3	3	3	3
a	3	0	3	0	3	0	3
k	3	3	0	3	3	3	3
e	3	3	3	3	3	3	3
d	3	3	3	3	0	3	3
a	3	0	3	0	3	0	3

	n	a	k	a	d	a	i
t	3	7	11	15	19	23	27
a	7	3	7	8	12	13	17
k	11	7	3	7	11	15	16
e	15	11	7	6	10	14	18
d	19	15	11	10	6	10	14
a	23	16	15	11	10	6	10

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

アルゴリズム



d へのルートは3種類

aまでの距離＋斜め移動のペナルティ＋不一致ペナルティ
bまでの距離＋下移動のペナルティ＋不一致ペナルティ
cまでの距離＋右移動のペナルティ＋不一致ペナルティ
の内の最短距離をdに書き込む

移動のペナルティ
横だけ1字ずらす 1
縦だけ1字ずらす 1
同時に1文字移動 0

不一致のペナルティ
文字が一致→0
文字が不一致→3

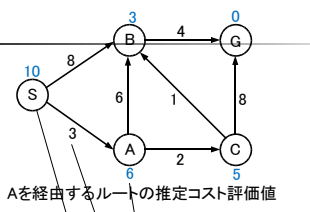
DPマッチングの応用

- DPマッチングの探索空間を制限し、探索時間を削減する方法
  - ビームサーチ（最適解は保証されない）
  - A*アルゴリズム（最適解は保証される）
- HMM（隠れマルコフモデル）とビタビアルゴリズム
  - 音声認識手法の主流

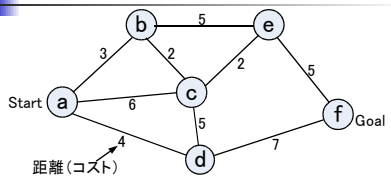
A*アルゴリズム 最短経路探索問題

- 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

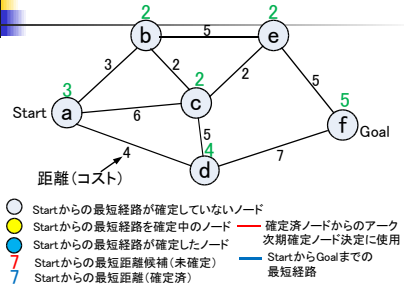
まずはAアルゴリズム



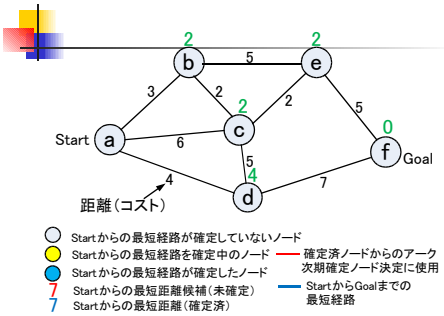
A*アルゴリズム



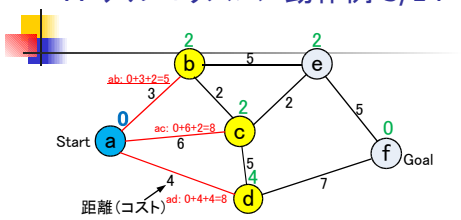
A*アルゴリズム　動作例 1/14



A*アルゴリズム　動作例 2/14

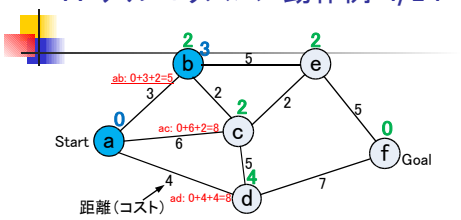


A*アルゴリズム　動作例 3/14



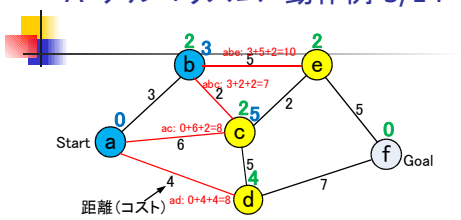
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 4/14



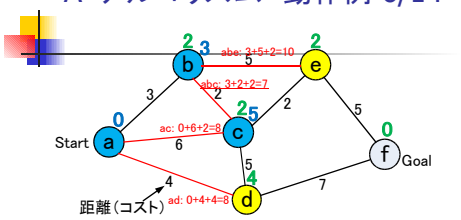
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 5/14



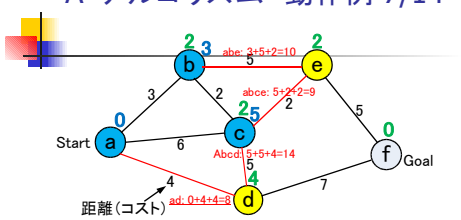
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 6/14



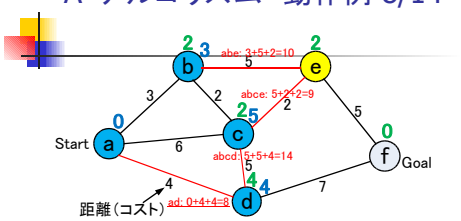
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 7/14



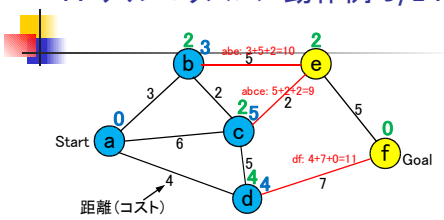
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 8/14



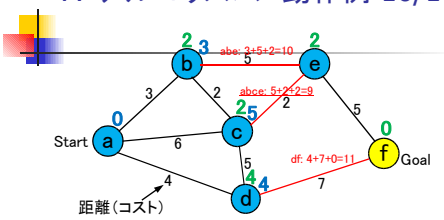
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 9/14



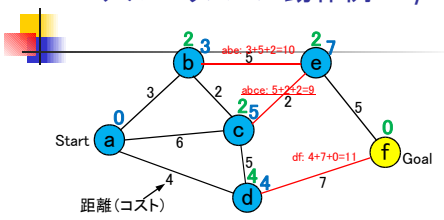
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 10/14



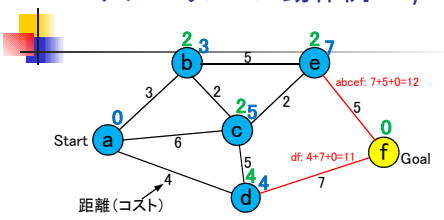
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 11/14



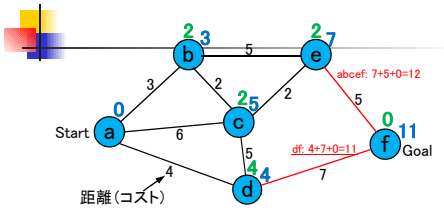
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 12/14



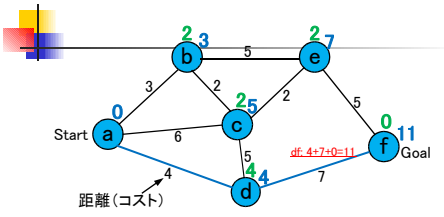
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

A*アルゴリズム　動作例 13/14



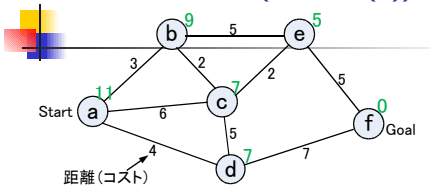
Startからの最短経路が確定していないノード
Startからの最短経路を確定中のノード
Startからの最短経路が確定したノード
Startからの最短距離候補（未確定）
Startからの最短距離（確定済）
確定済ノードからのアーク
次期確定ノード決定に使用
StartからGoalまでの
最短経路

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

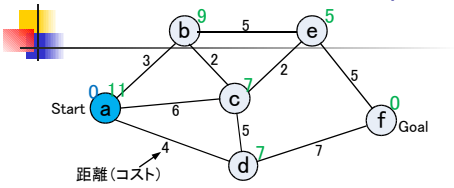
A*アルゴリズムその2 (最良のh(n)) 1/6



○ 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
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
次期確定ノード決定に使用
— StartからGoalまでの
最短経路

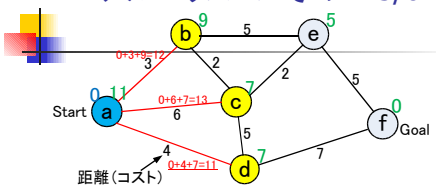
A*アルゴリズム その2 2/6



○ 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
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
次期確定ノード決定に使用
— StartからGoalまでの
最短経路

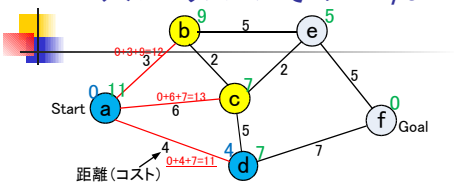
A*アルゴリズム その2 3/6



○ 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
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
次期確定ノード決定に使用
— StartからGoalまでの
最短経路

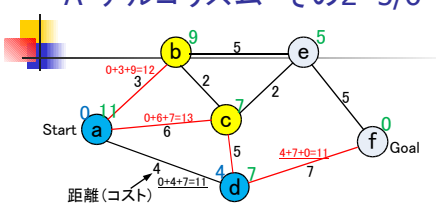
A*アルゴリズム その2 4/6



○ 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
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
次期確定ノード決定に使用
— StartからGoalまでの
最短経路

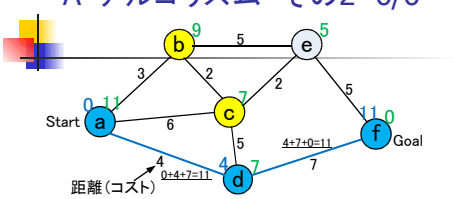
A*アルゴリズム その2 5/6



○ 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
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
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 6/6



○ 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
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
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，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≦h*(n)≦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)

来週は中間試験

- 試験範囲は今日の授業まで
  - スタック
  - 文脈自由文法
  - 構文解析
  - CYK法
  - （トップダウンチャート法）
  - ダイクストラ法
  - DPマッチング
  - A*アルゴリズム
- 試験問題の傾向はWebに掲載している去年の試験問題を見てください