

mi-4. 発見的探索

(人工知能シリーズ)

<https://www.kkaneko.jp/cc/mi/index.html>

金子邦彦



今日の内容

- 正解の**探索**において, **発見的探索**を行う

アウトライン

4-1 パスと木

4-2 グラフの中の木

4-3 グラフと全域木

4-4 探索

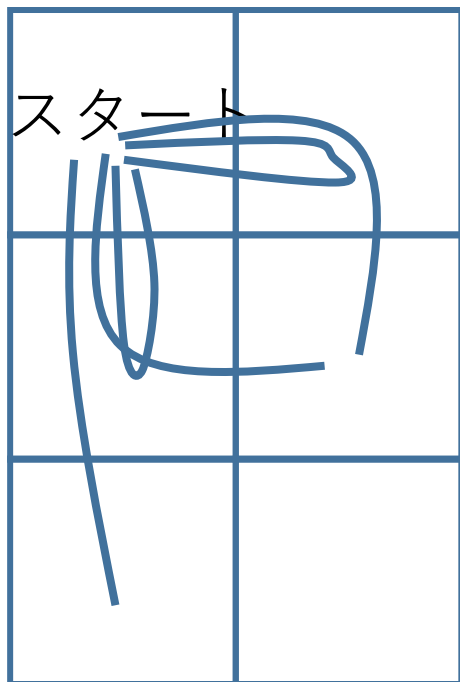
4-5 横型探索

4-6 A* 法

4-1 パスと木

パスの例

- **総当たり**では、すべての経路 (**パス**) を試す



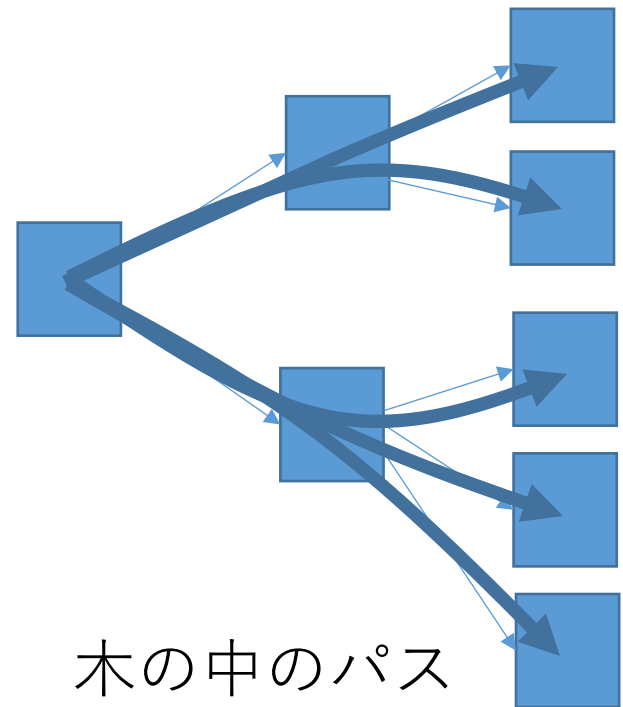
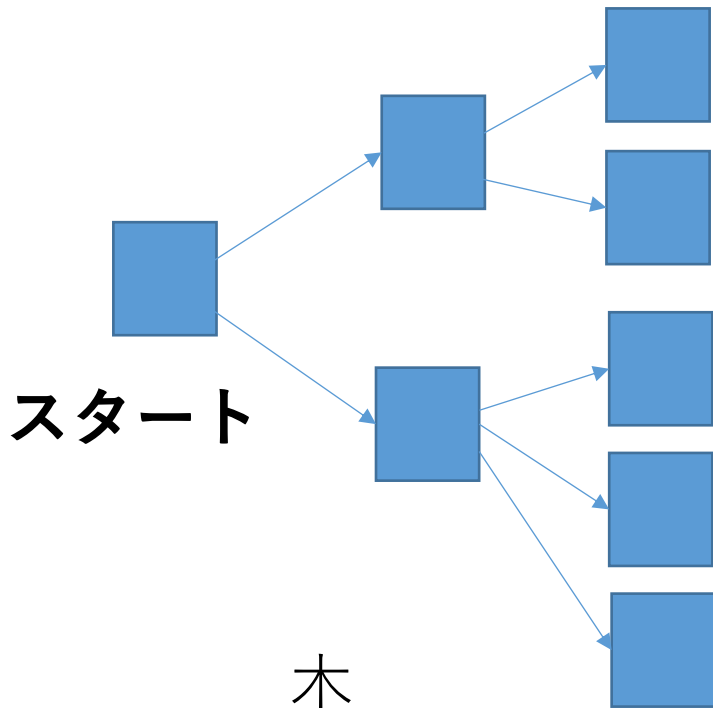
5つのパス

- ルールの並び : 1, 2
- ルールの並び : 1, 3
- ルールの並び : 3, 1
- ルールの並び : 3, 3
- ルールの並び : 3, 4

パスと木

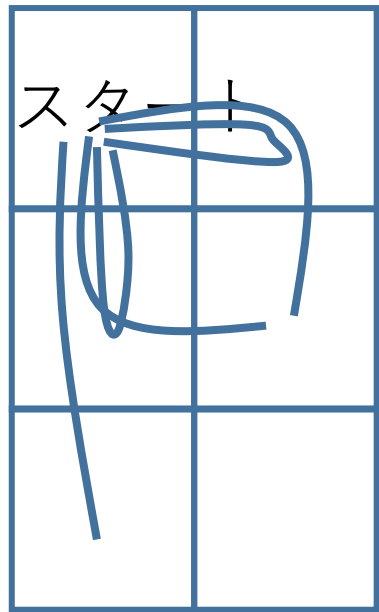
• **木**とは**スタート**が同じであるような**パス**の集まり。

※ 木では、パスが合流するようなことはない



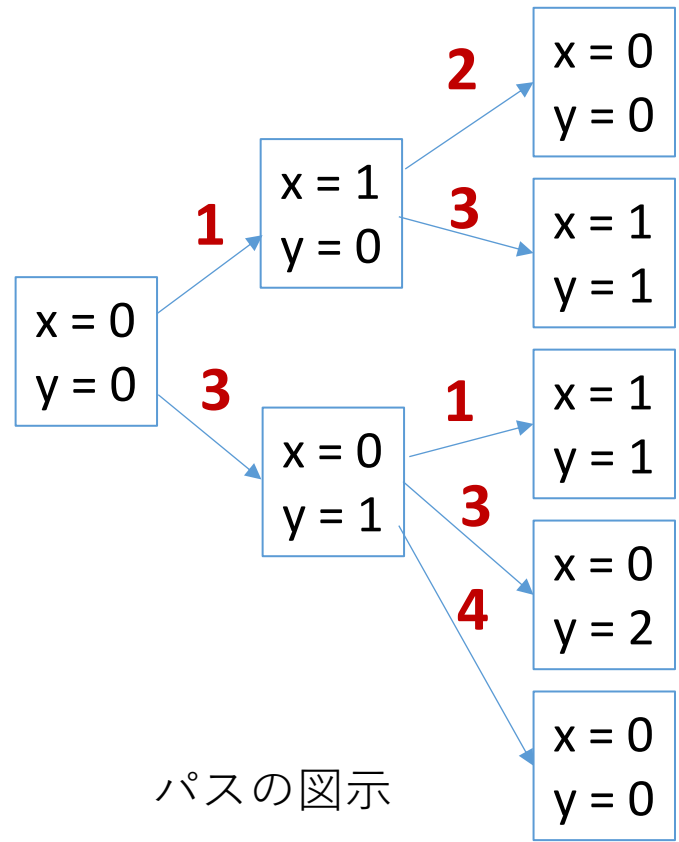
総当たりでのパスの図示

- ・ **総当たり**では、すべての経路（**パス**）を試す



5つのパス

- ・ ルールの並び : **1, 2**
- ・ ルールの並び : **1, 3**
- ・ ルールの並び : **3, 1**
- ・ ルールの並び : **3, 3**
- ・ ルールの並び : **3, 4**

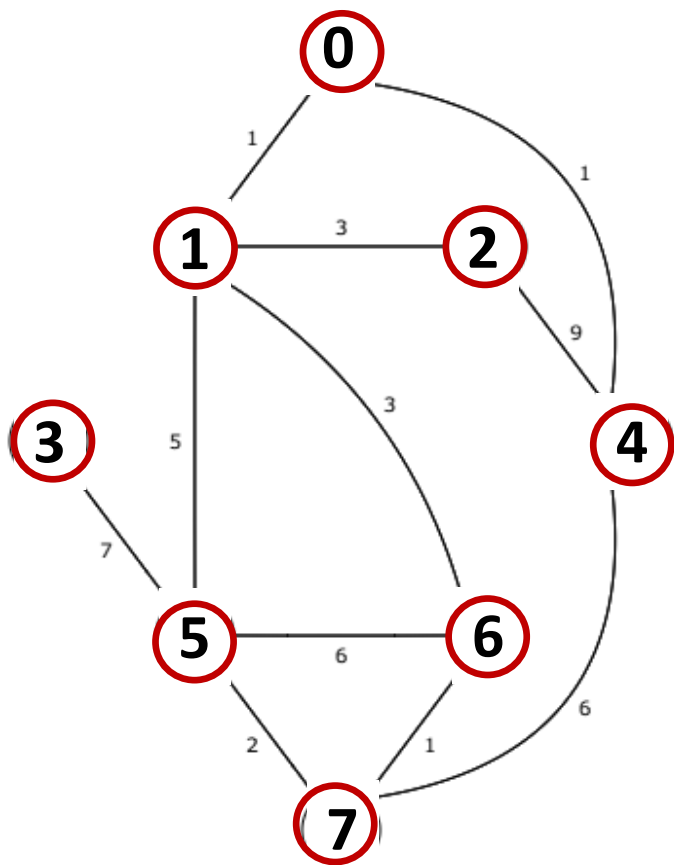


パス長 2 の経路（パス）をすべて試す

パスの図示

4-2 グラフの中の木

グラフ

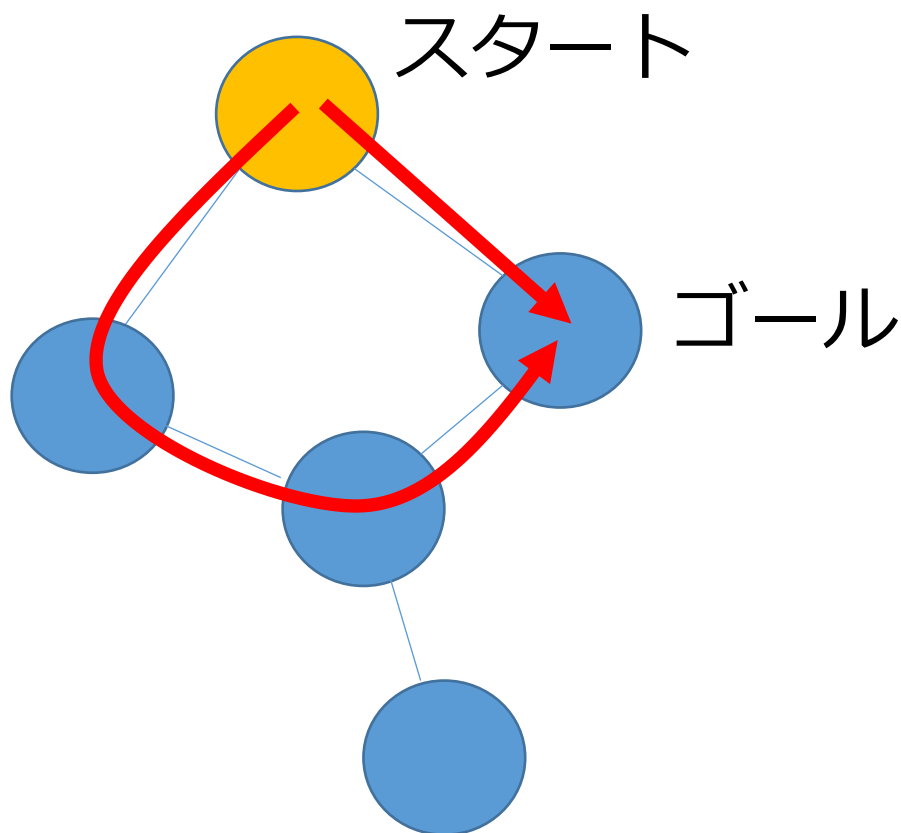


○ ノード
線 エッジ

用途

- ・道路のつながり具合
 - ・バス路線
- など

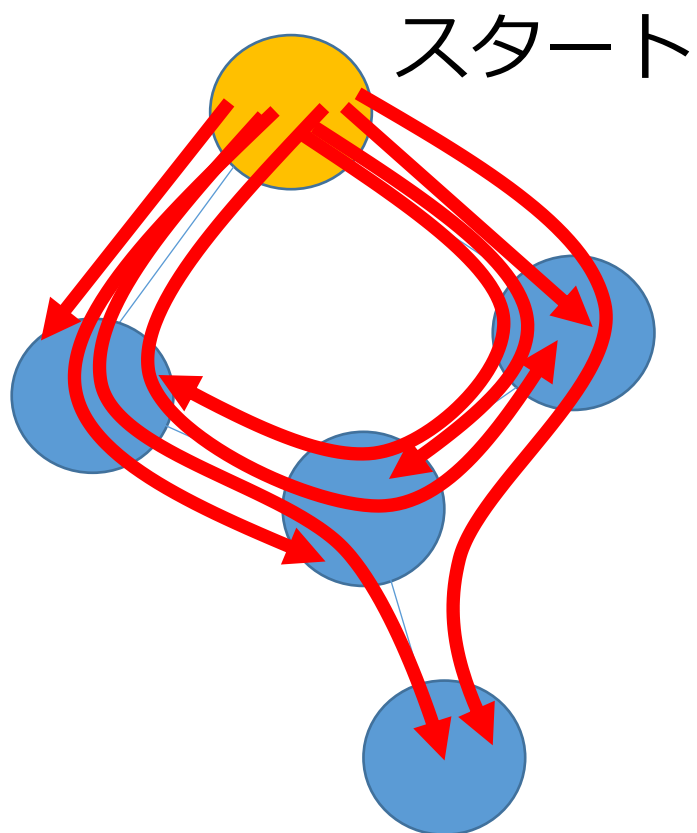
グラフとパス



○ ノード
線 エッジ

グラフの中で、
スタートと**ゴール**を指定
→ **パスは複数あり得る**
(分岐と合流)

グラフとパス

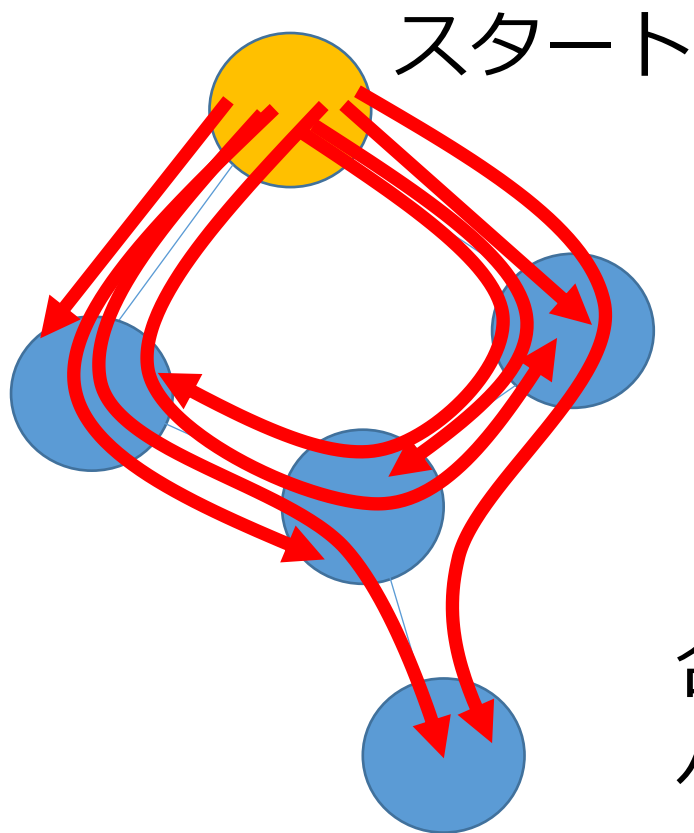


○ ノード
線 エッジ

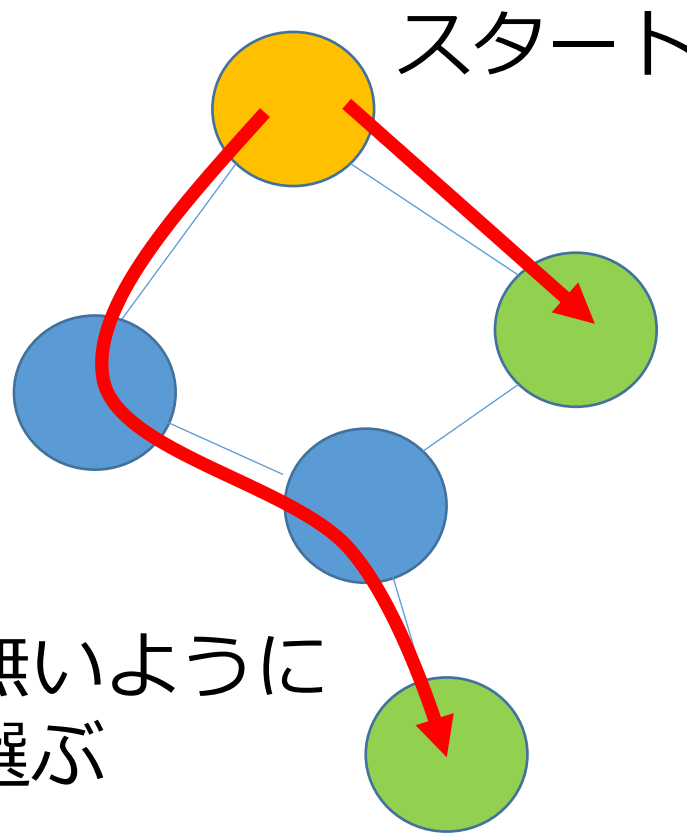
グラフの中で,
スタートを指定
→ **パスは複数あり得る**
(分岐と合流)

グラフと木

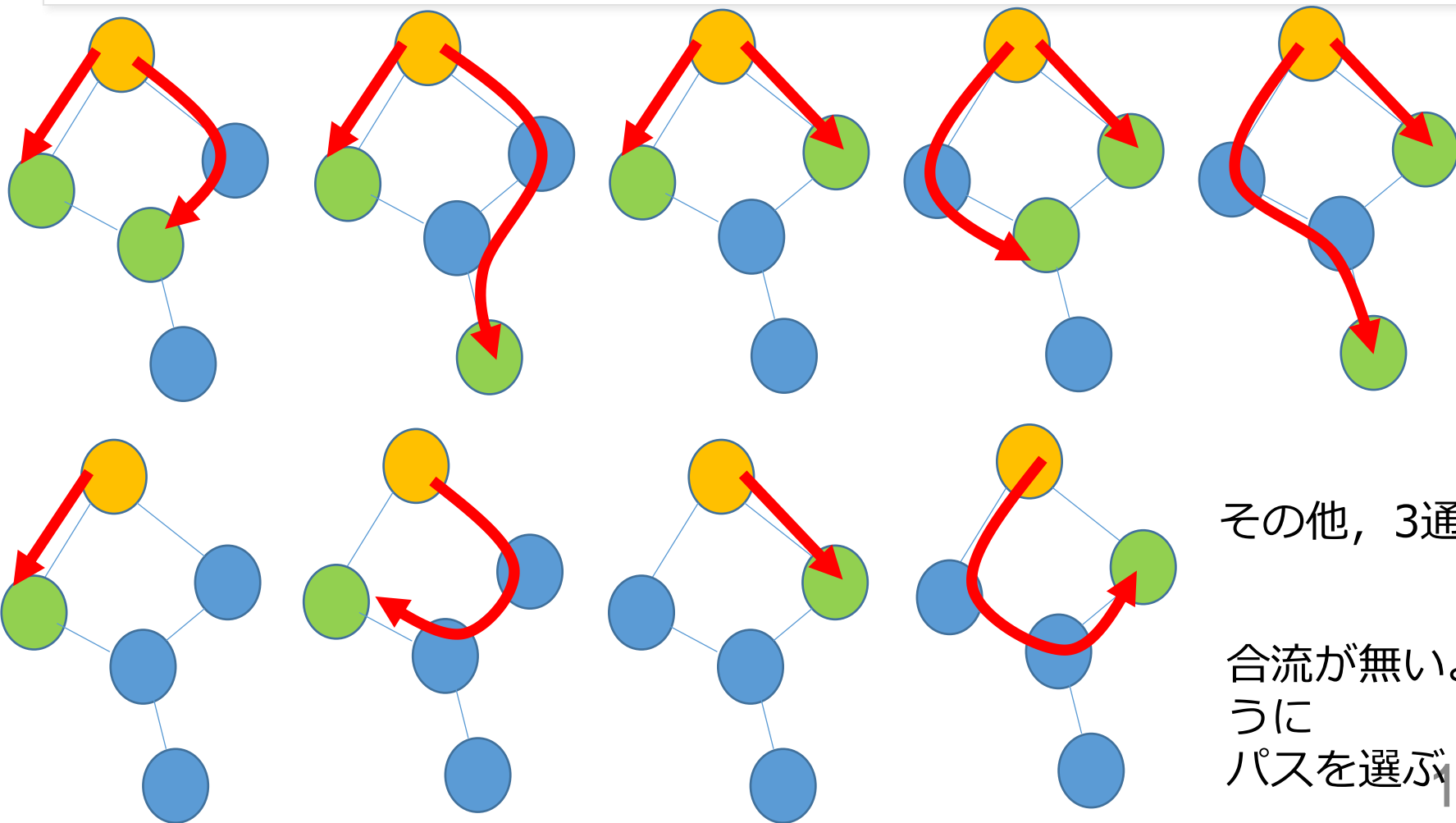
グラフの中で、**スタート**を指定し、**木**を構成する



合流が無いようにパスを選ぶ



グラフと木

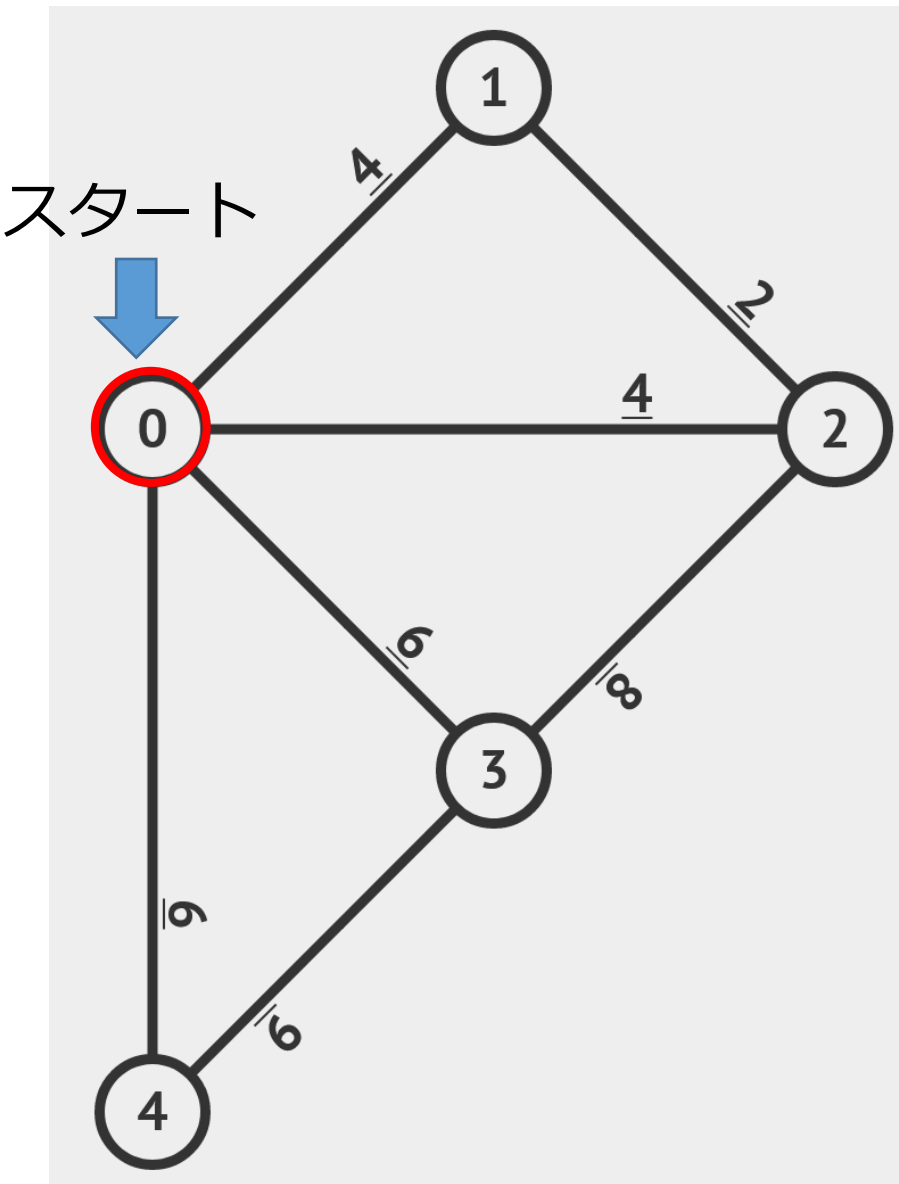


おわりに

- **人工知能**では、さまざまな情報を扱う
- **パス, 木, グラフ**は、情報を見通し良く扱えるための考え方
- **総当たり**では、すべての**パス**を試すが、パスの情報を見通し良く扱うに**木**は有効

4-3 グラフと全域木

グラフ内のノードをすべて含む木 (全域木)



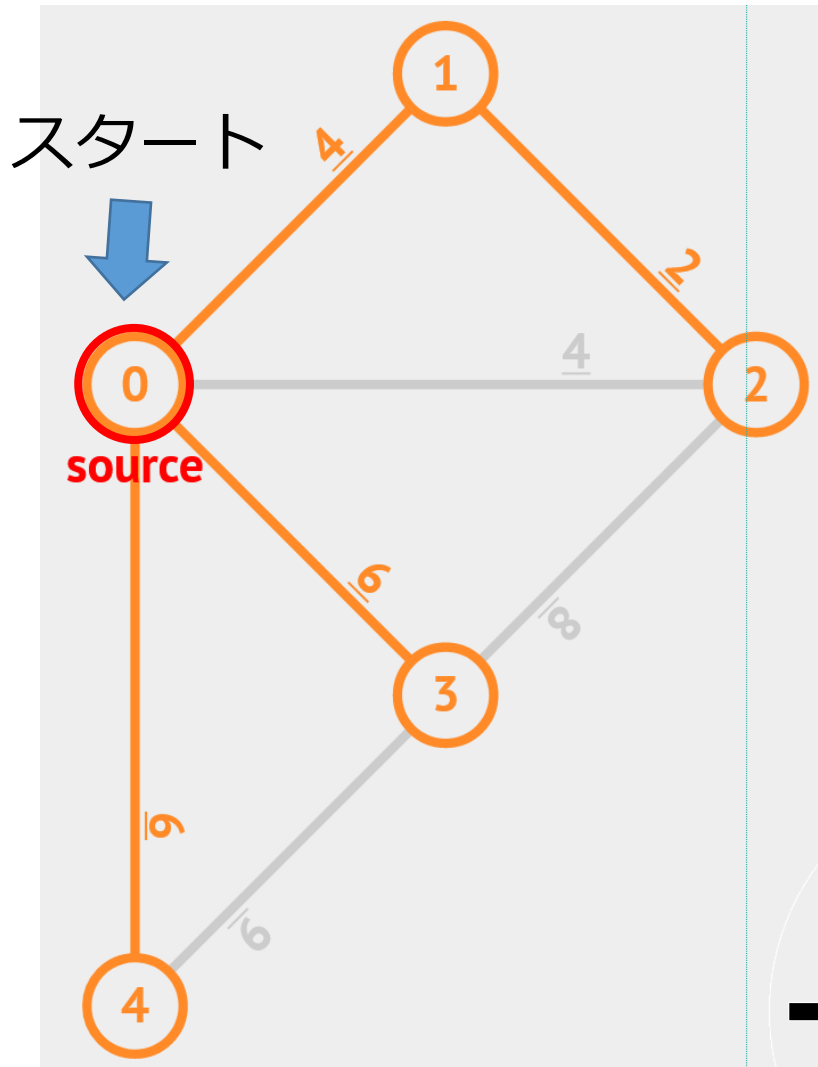
最短経路探索 (カーナビなど)

○ ノード
線 エッジ

このグラフでは、**エッジ**ごとの
移動コストが分かっている

すべてのノードをつなぐ。
コストが最小になるようにする。
例) 水道管, 電気配線

グラフ内のノードをすべて含む木（全域木）



最短経路探索（カーナビなど）

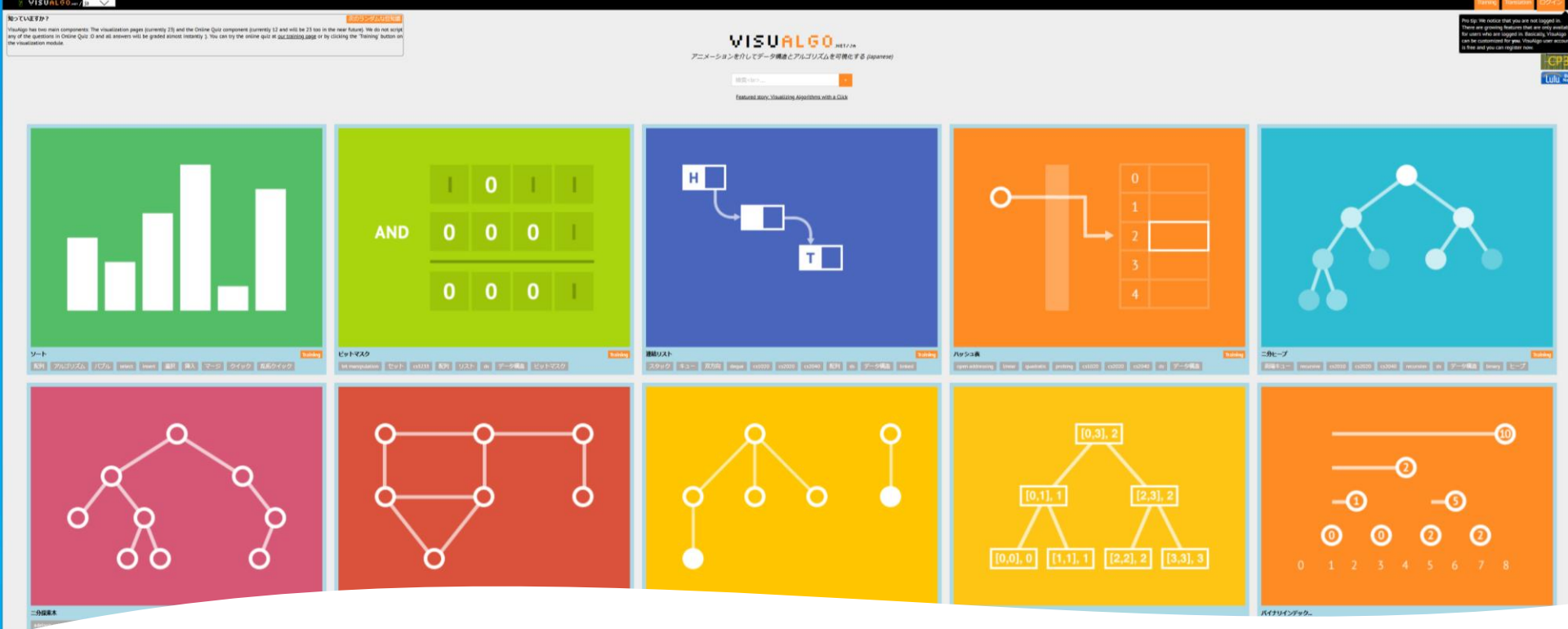
○ ノード
線 エッジ

このグラフでは、**エッジ**ごとの
移動コストが分かっている

すべてのノードをつなぐ。
コストが最小になるようにする。
例) 水道管, 電気配線

グラフの中の**木**
(赤線で**木**の**パス**を示す)

<https://visualgo.net/ja> よりコピー



ビジュアルに、グラフや木などの情報技術を無料で学習できるオンラインサービス

VisuAlgo

① ウェブブラウザで次の URL を開く

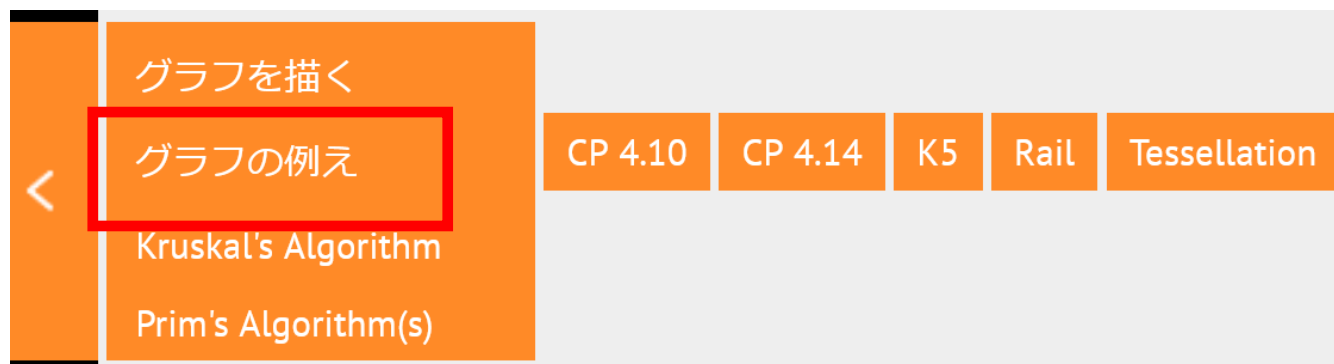
<https://visualgo.net/ja>

② メニューが表示されるので確認

① 「最小全域木」をクリック



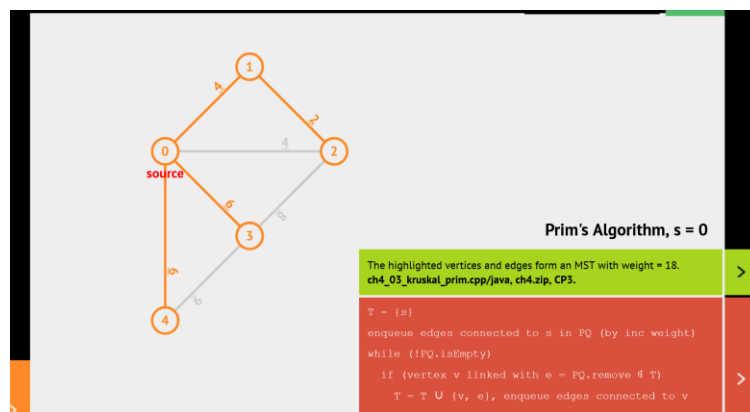
② 「グラフの例え」でグラフを選ぶ



③ 「Prim's Algorithm(s)」を選び，スタートとする
ノードの番号を設定。「行く」をクリック。



④ アニメーションで，算出過程が表示されたのち，
結果が表示される



4-4 探索

探索の考え方

• 総当たり

正解に至る経路（パス）を**探索**するために、すべての経路（パス）を試す

※ 「正解に至るパスは必ず見つける」、 「正解に至るパスはすべて見つける」ための技術

• 単純な**探索**

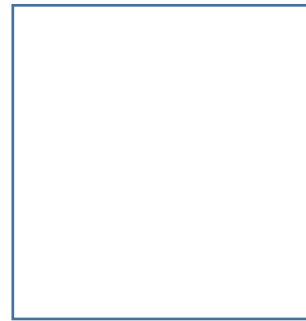
正解に至る経路（パス）が1つ見つかった時点で**探索を打ち切る**

2つの水差し

- 水差し① 大きさ**4**
- 水差し② 大きさ**3**



水差し①

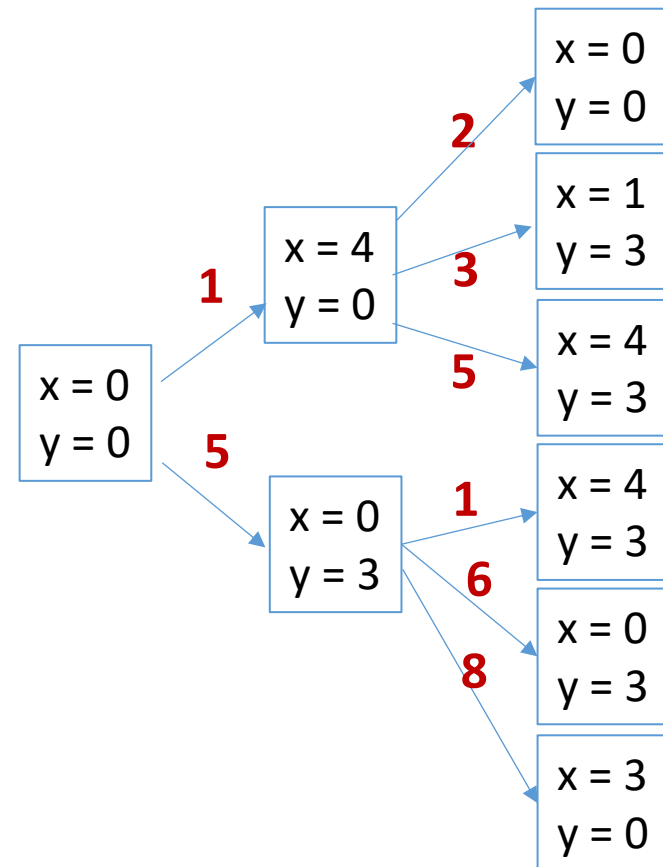


水差し②

総当たり

- **総当たり**では, すべての経路 (**パス**) を試す

(1, 2)	0	0
(1, 3)	1	3
(1, 5)	4	3
(5, 1)	4	3
(5, 6)	0	3
(5, 8)	3	0



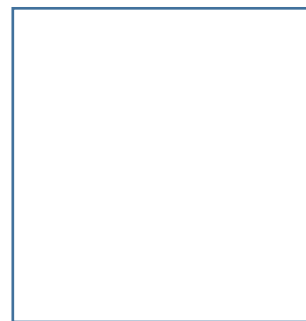
パス長 2 の経路 (パス) をすべて試す
※ パソコンでの操作手順は, 第3回授業
の資料

ゴールの例

- どちらの水差しでもよいから、量「1」の水が欲しい

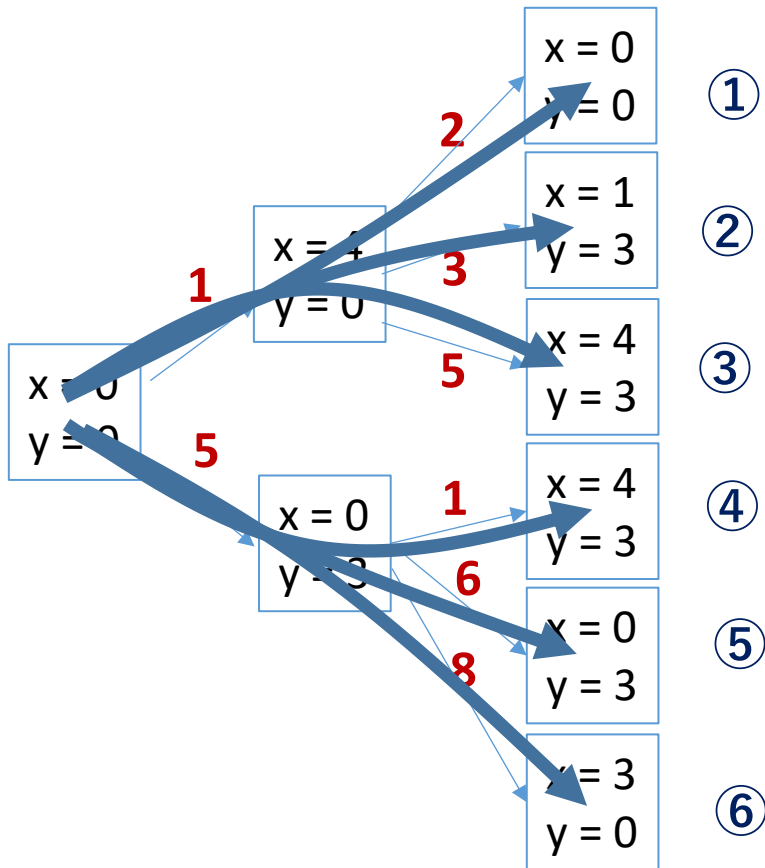


水差し①



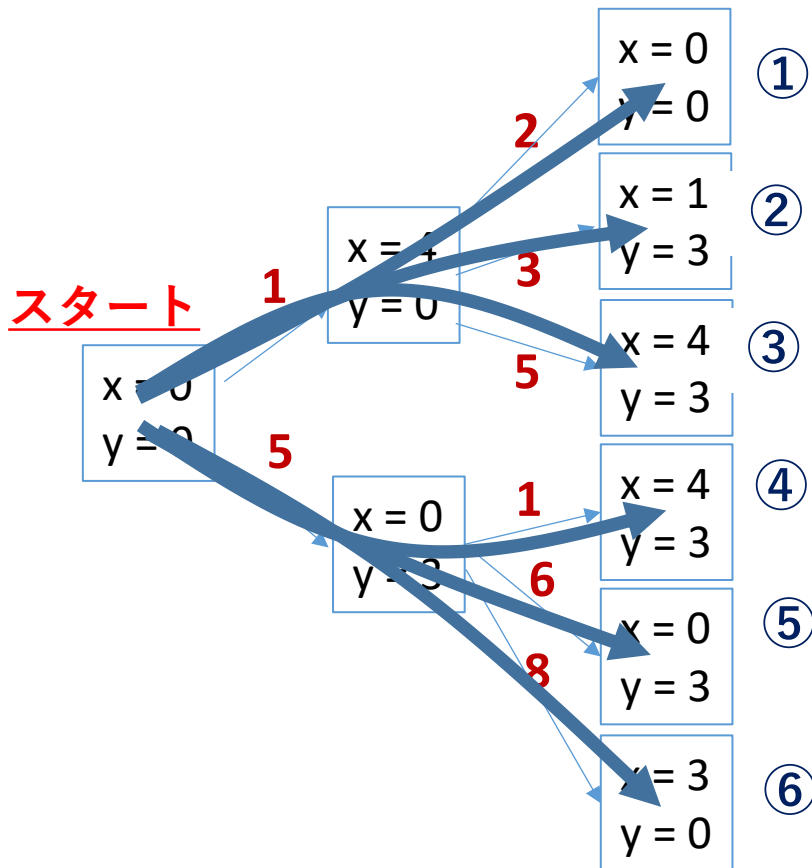
水差し②

探索の打ち切り



- 量「1」が欲しいとする
 - **総当たり**を始めるが、「1」が見つかった時点で打ち切る
- ①, ② で **打ち切り**

探索とは



- **探索**とは, **木**の中の**スタート**の場所と, **ゴール状態**を指定して, **パス**を探すこと

ゴール状態の例

$$x = 1 \text{ または } y = 1$$

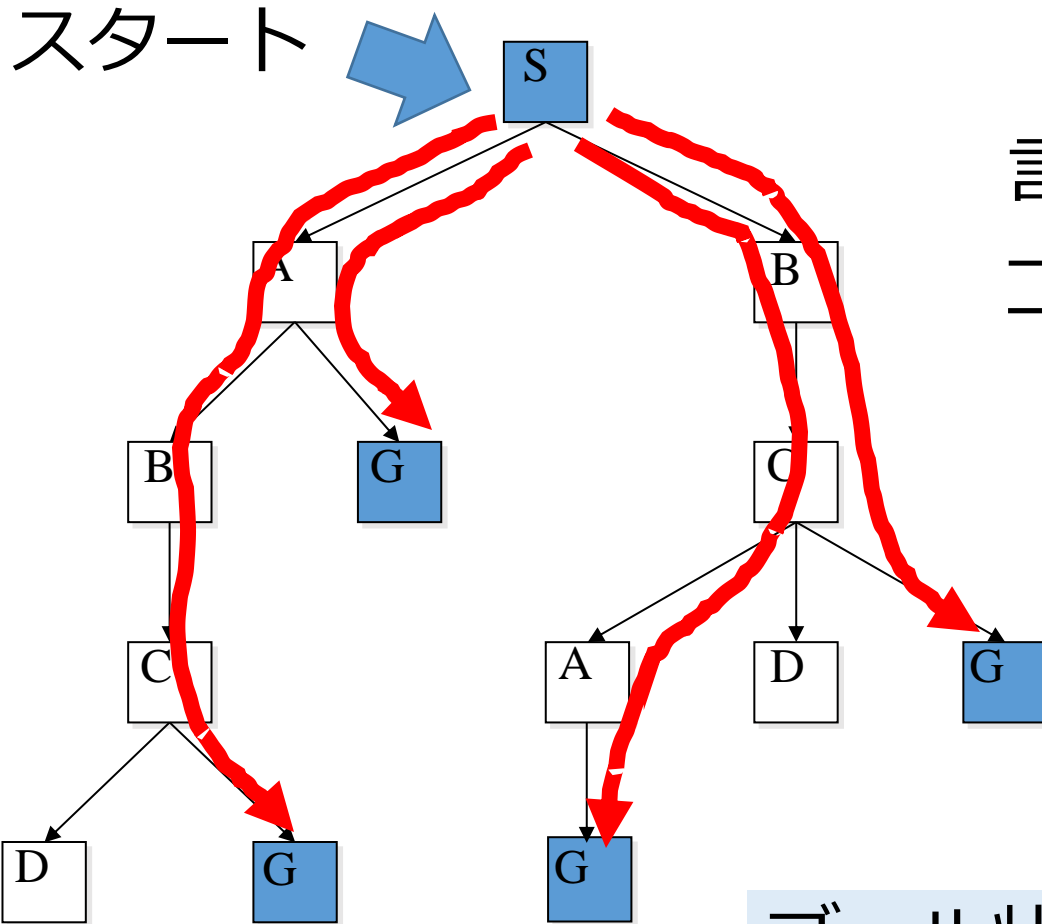
探索をコンピュータに解かせることは、
人工知能の一種である
(コンピュータが知性を示している)

探索で解けそうな問題の例



- ルート発見
- 部品や配線の配置
- ロボットの誘導
- 小さなパズル (3×3の碁盤でオセロゲームなど)

- **探索**とは、**木**の中の**スタート**の場所と、**ゴール**の**状態**を指定して、**パス**を探ること



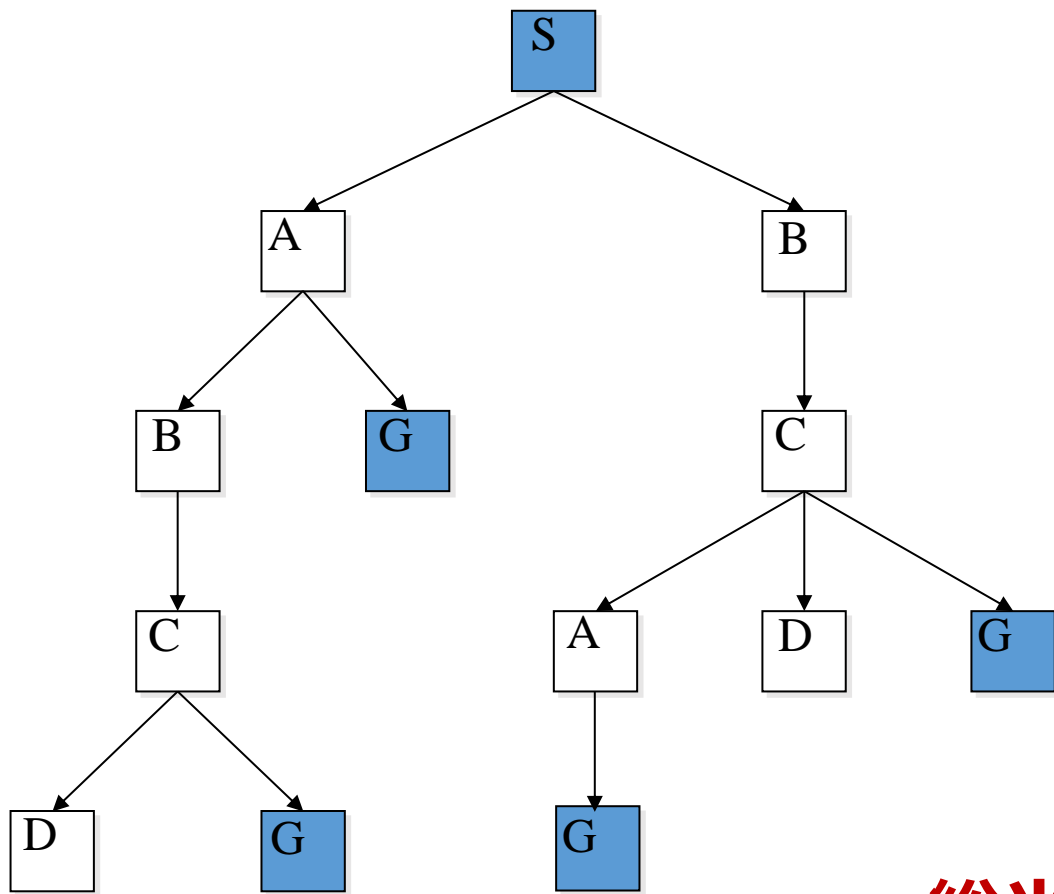
「ゴールの場所」とは
言っていない。
ゴールは**複数ありえる**

パスを1つ見つければよい
という場合は、総当たりを
途中で打ち切る

ゴール状態：G

4-5 横型探索

総当たりの探索



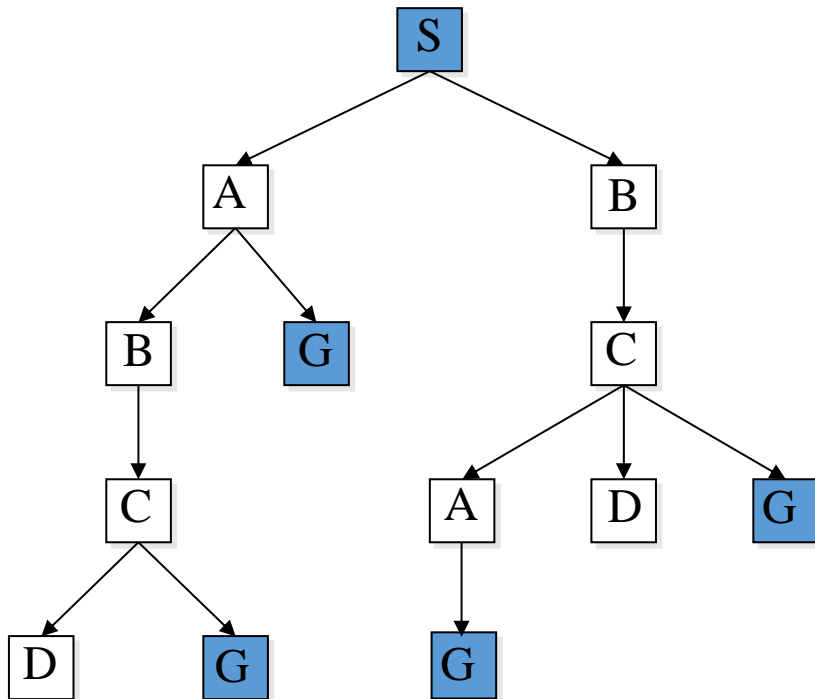
パスは6つ考える

1. S A B C D
2. S A B C G
3. S A G
4. S B C A G
5. S B C D
6. S B C G

総当たりでは、
すべての経路 (**パス**) を
試す

「パスの長さの短いもの」を先に探す

- ・ 利点： 最も短いものが求まるので嬉しい場合がある
- ・ 欠点： 短い順にそろえる手間を要する



3. **S A G**

5. **S B C D**

6. **S B C G**

1. **S A B C D**

2. **S A B C G**

4. **S B C A G**

のような順で探索

おわりに

- 正解に至る経路（**パス**）が1つ見つかった時点で**探索を打ち切る**とき、
- 使用している**総当たり**の種類によって、見つかるまでの速さが変わる
- 次の資料では、横型探索でない別の種類（A*法）を説明する

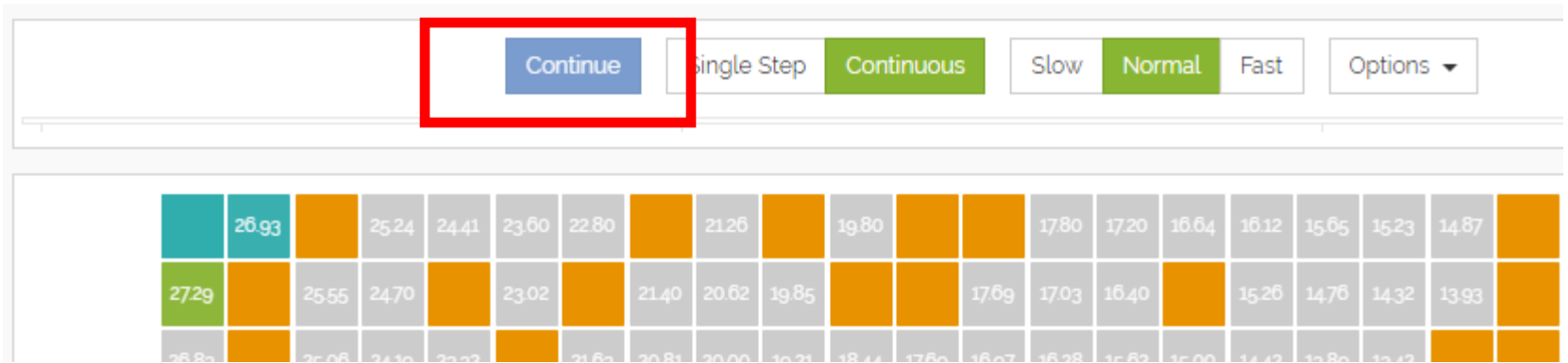
4-6 A* 法

①次の URL を開く

<http://www.algomatic.com/algorithm/a-star-maze-solver>

② A* 法のアニメーションが始まるので確認

はじまらないときは「Continue」をクリック



Continue Single Step Continuous Slow Normal Fast Options ▾

26.93		25.24	24.41	23.60	22.80		21.26		19.80			17.80	17.20	16.64	16.12	15.65	15.23	14.87		
27.29		25.55	24.70		23.02		21.40	20.62	19.85			17.69	17.03	16.40		15.26	14.76	14.32	13.93	
26.82		25.06	24.19	23.22		21.62	20.81	20.00	19.21	18.44	17.69	16.97	16.28	15.62	15.00	14.42	13.89	13.42		

③ A* 法による探索結果を確認

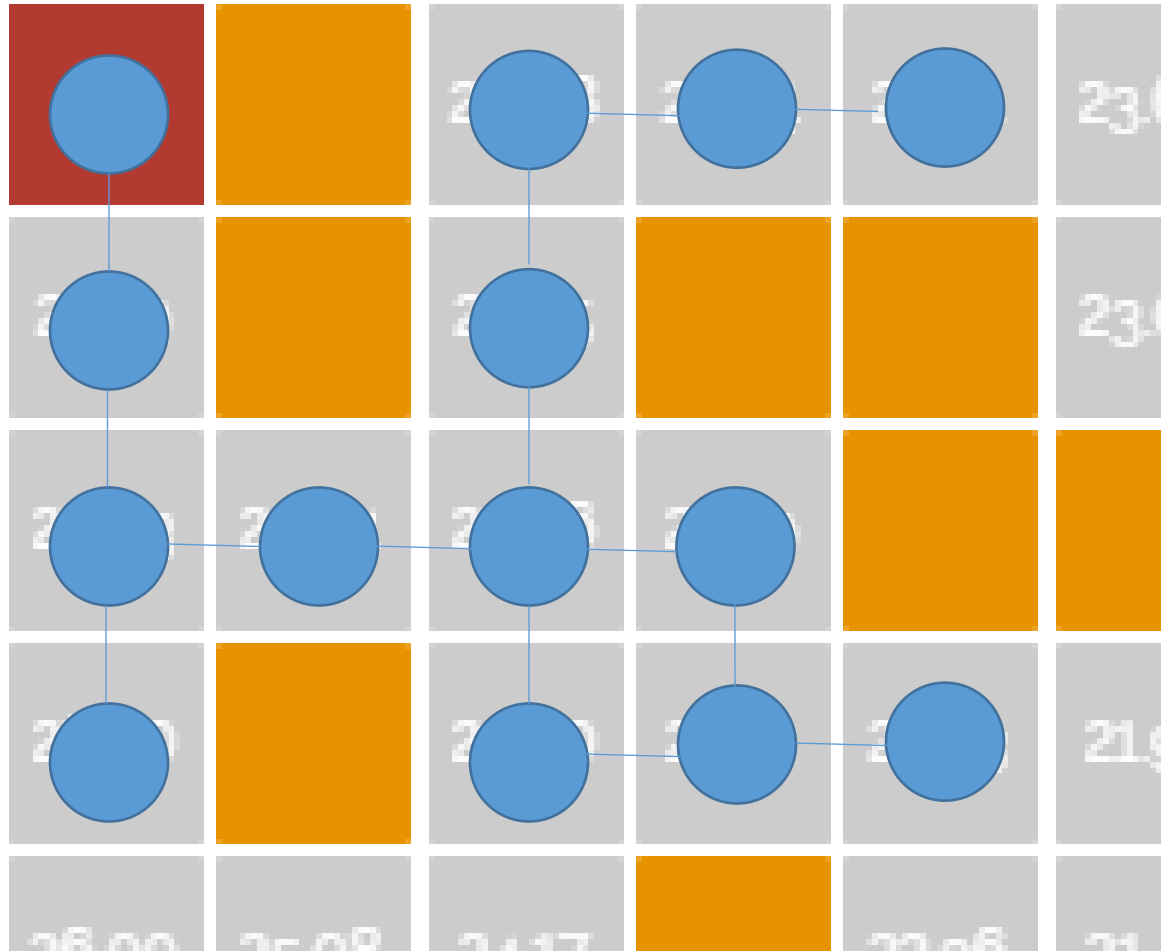
スタートからゴールまでのパスが青の四角で示されるので確認

※ **緑**は、**通らないことにした**方向

26.93	25.24	24.41	23.60	22.80	21.26	19.80	17.80	17.20	16.64	16.12	15.65	15.23	14.87	14.32	14.00						
27.29	26.55	24.70	23.02	21.40	20.62	19.85	17.69	17.03	16.40	15.26	14.76	14.32	13.93	13.15	13.04	13.00					
26.83	25.06	24.19	23.32	21.63	20.81	20.00	19.21	18.44	17.69	16.97	16.28	15.62	15.00	14.42	13.89	13.42	12.37	12.17	12.04	12.00	
26.40	25.50	24.60	23.71	21.95	20.25	17.80	16.28	16.56	14.87	13.60	12.53	11.40	11.18	11.05	11.00						
25.08	24.17	23.26	22.36	21.47	20.59	18.87	18.03	17.20	16.40	15.62	14.14	13.45	12.81	12.21	11.66	10.77	10.44	10.20	10.05	10.00	
25.63	24.70	23.77	22.85	21.93	19.24	18.36	17.49	16.64	15.81	14.21	13.45	12.04	11.40	10.82	10.30	9.85	9.49	9.22	9.06	9.00	
25.30	23.41	22.47	21.54	19.70	18.79	17.89	16.12	14.42	13.60	12.04	11.31	10.00	9.43	8.94	8.06	8.00					
25.00	24.04	23.09	20.25	19.31	18.38	17.46	16.55	15.65	14.76	13.89	13.04	12.21	11.40	10.63	9.90	9.22	7.62	7.07	7.00		
23.77	21.84	20.88	18.97	18.03	17.09	16.16	15.23	14.32	13.42	12.53	11.66	10.82	10.00	9.22	8.49	7.81	7.21	6.32			
24.52	23.54	22.56	20.62	19.65	18.68	17.72	16.76	14.87	13.93	11.18	10.30	9.43	8.60	7.07	6.40	5.83	5.39	5.10	5.00		
23.35	22.36	20.40	19.42	18.44	17.46	16.49	14.56	13.60	11.70	10.77	9.85	8.94	8.06	7.21	5.66	5.00	4.47	4.12			
24.19	22.20	21.21	20.22	19.24	18.25	17.26	16.28	15.30	13.34	12.37	11.40	10.44	9.49	8.54	7.62	6.71	4.24	3.61	3.00		
24.08	23.09	22.09	21.10	20.10	19.10	18.11	17.12	16.12	14.14	13.15	11.18	10.20	8.25	7.28	6.32	5.39	4.47	3.61	2.83	2.24	2.00
22.02	20.02	19.03	18.03	15.03	14.04	12.04	11.05	10.05	9.06	8.06	7.07	5.10	4.12	1.00							
24.00	23.00	22.00	21.00	20.00	18.00	17.00	16.00	15.00	9.00	8.00	7.00	6.00	5.00	3.00	2.00	1.00					

迷路の探索も木である

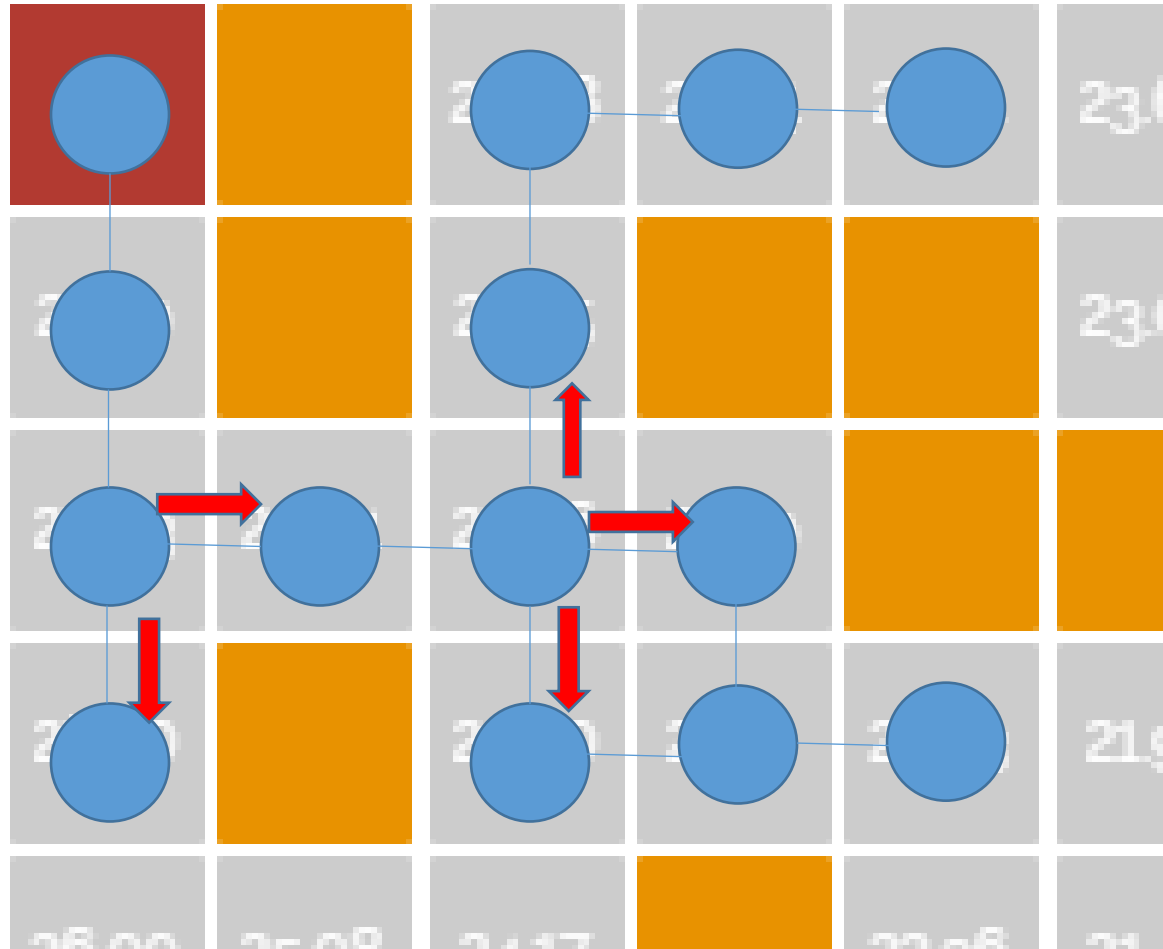
スタート



A* 法のルール



1. 分岐では, 「よりゴールに近い」方を**選ぶ**



A* 法のルール

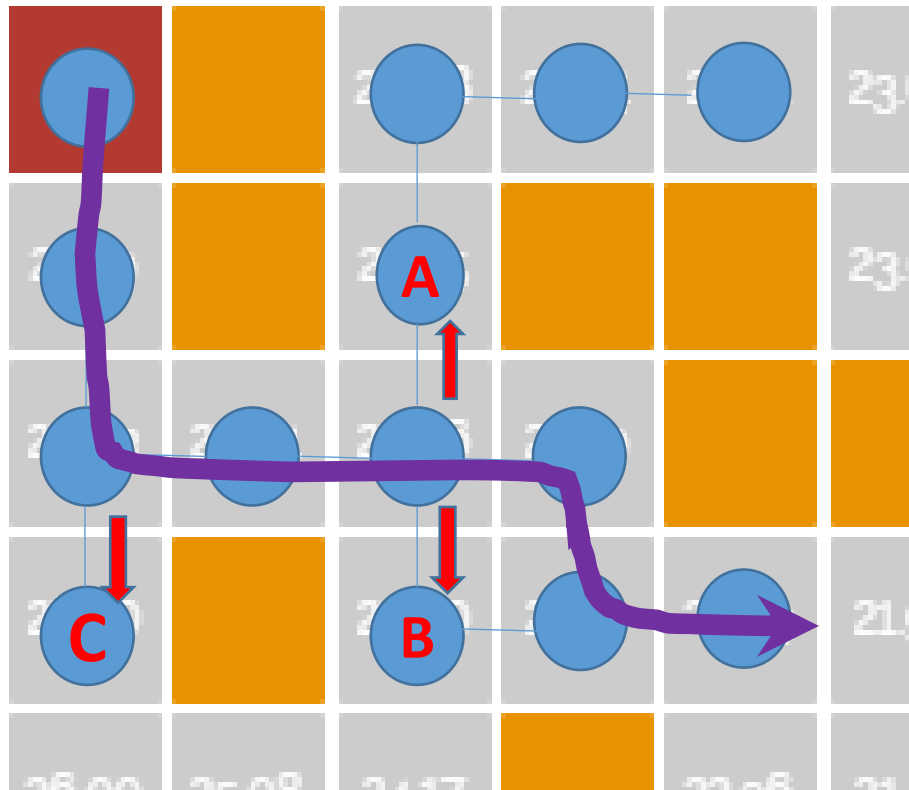
2. 行き止まり, 難所に来たら

今まで通った分岐の中で,

「スタートから分岐までにかかったコスト」と

「その分岐からもう1歩動くコスト」の合計が最小の

分岐まで戻る



A, B, C の中で
最も良いところに
進路を変える



こちらに行き止まり,
難所があったとする

おわりに

- **探索**とは、**木**の中の**スタート**の場所と、**ゴール状態**を指定して、**パス**を探すこと
- **探索**をコンピュータに解かせることは、**人工知能**の一種である
- **探索**には、いろいろな種類がある