木における1ラウンドボロノイゲームの後手の戦略

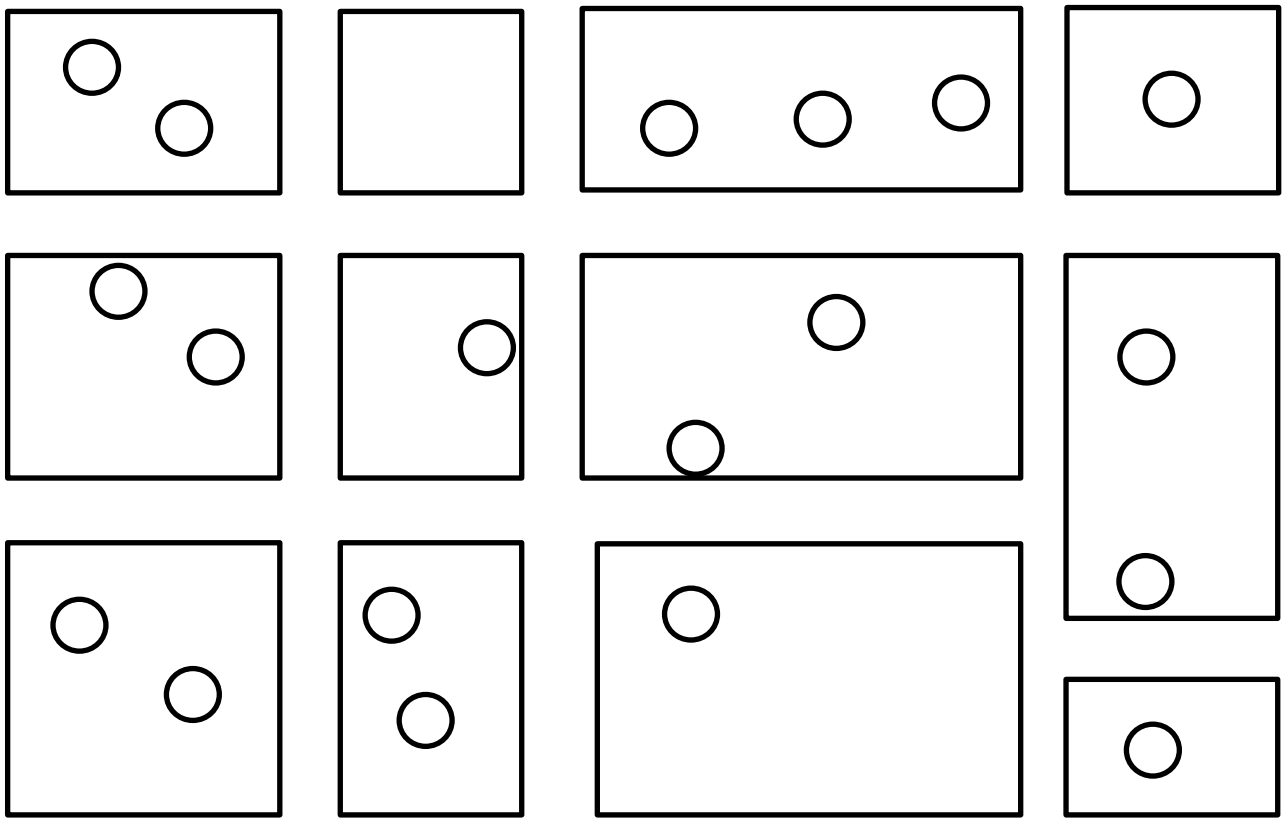
神戸大学大学院　**杉本晃弘（発表者）**

九州工業大学　斎藤寿樹

ボロノイゲームとは？

店舗出店シュミレーション

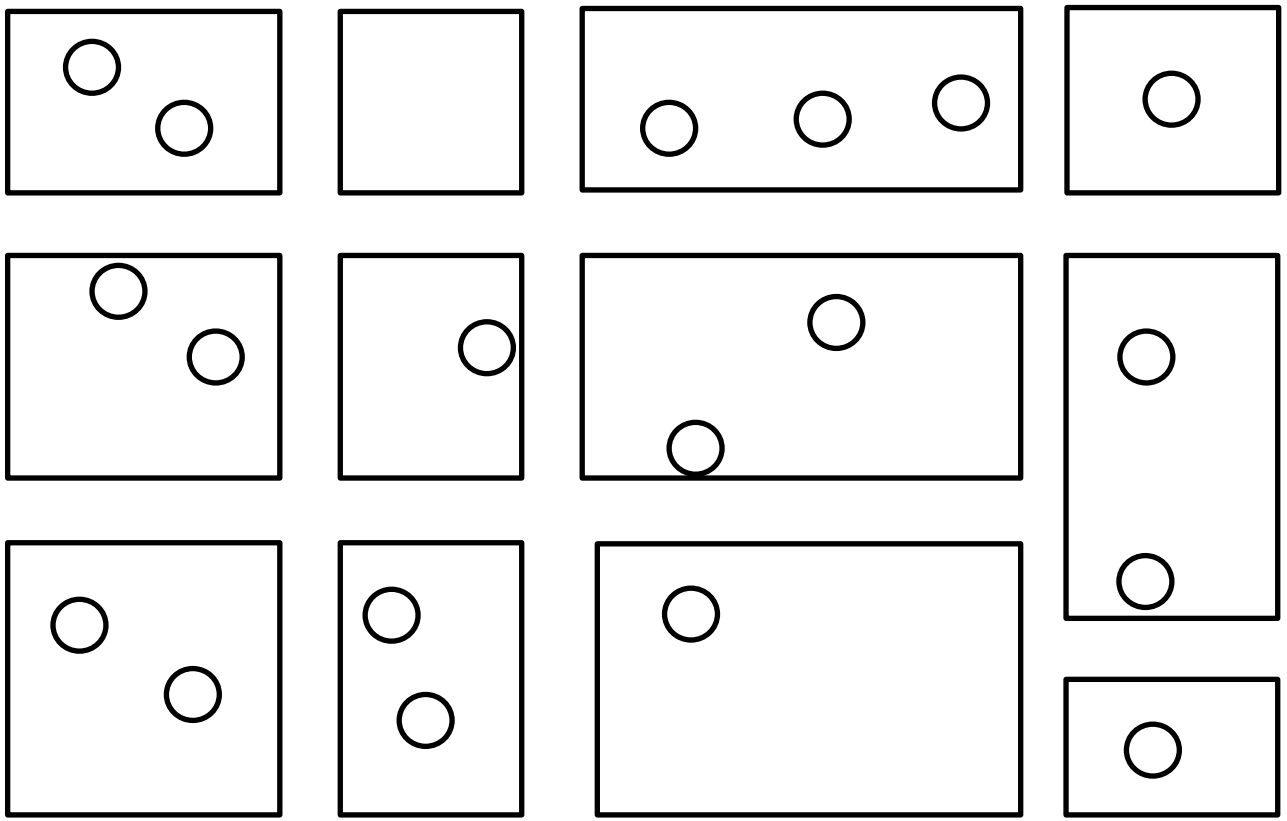
↓地図



○：民家

ボロノイゲームとは？

コンビニのL店とS店が交互に出店する。



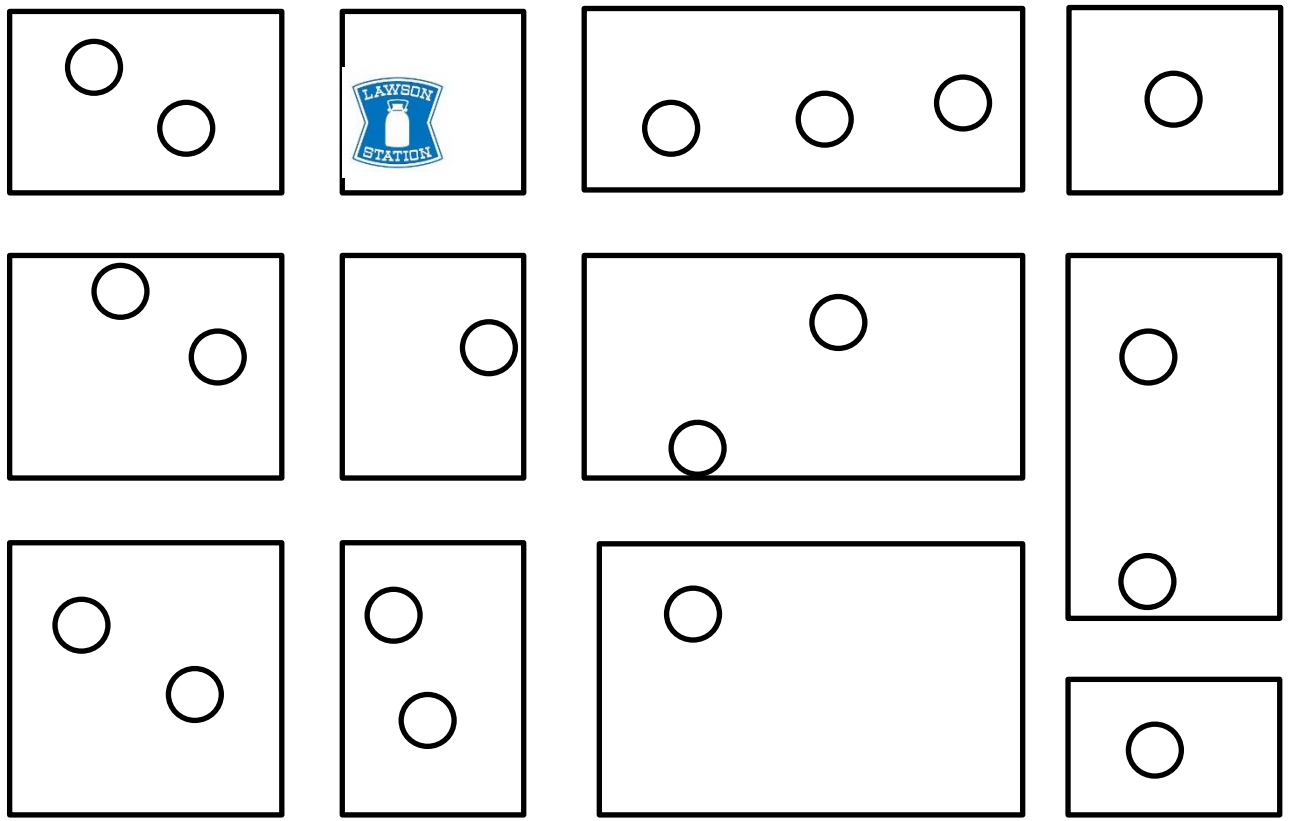
○：民家

：L店

：S店

ボロノイゲームとは？

コンビニのL店とS店が交互に出店する。



先手必勝!

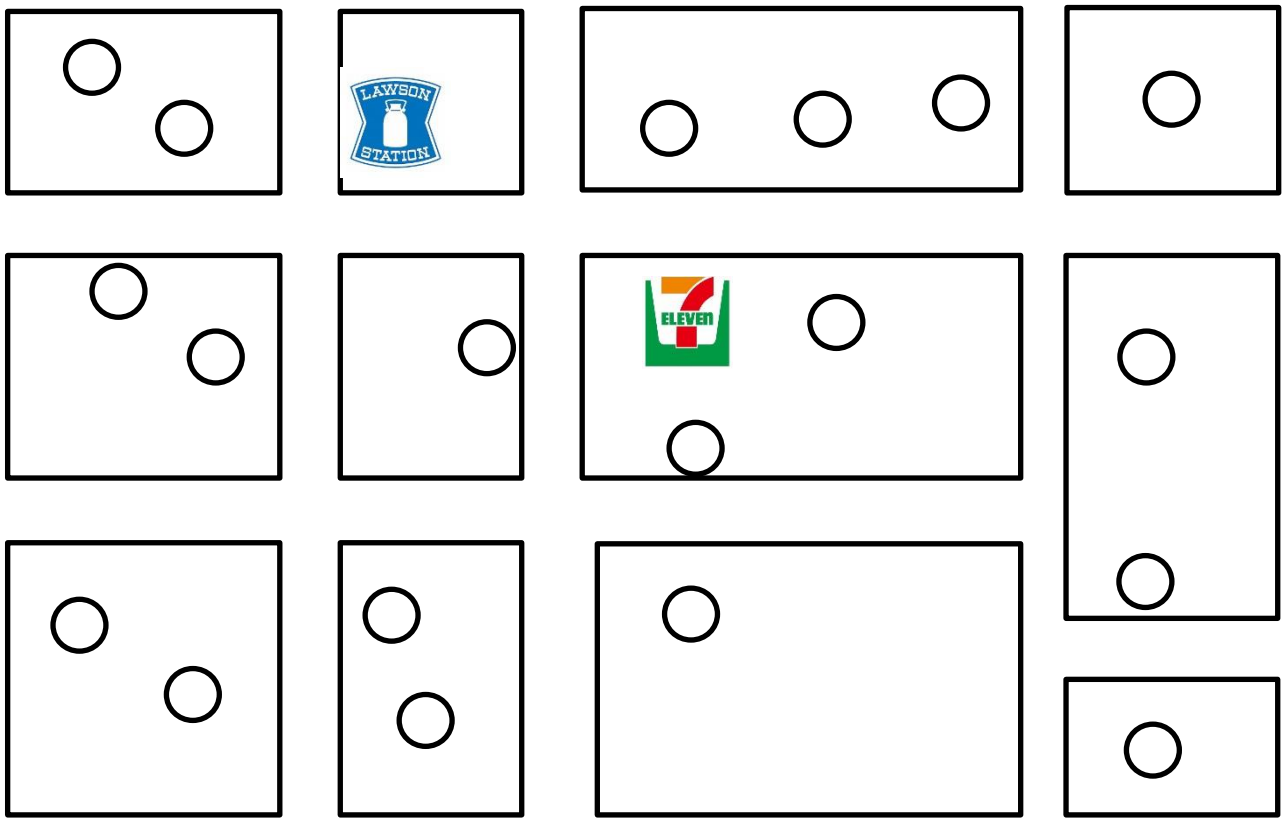
○：民家

：L店

：S店

ボロノイゲームとは？

コンビニのL店とS店が交互に出店する。



このへんかな？

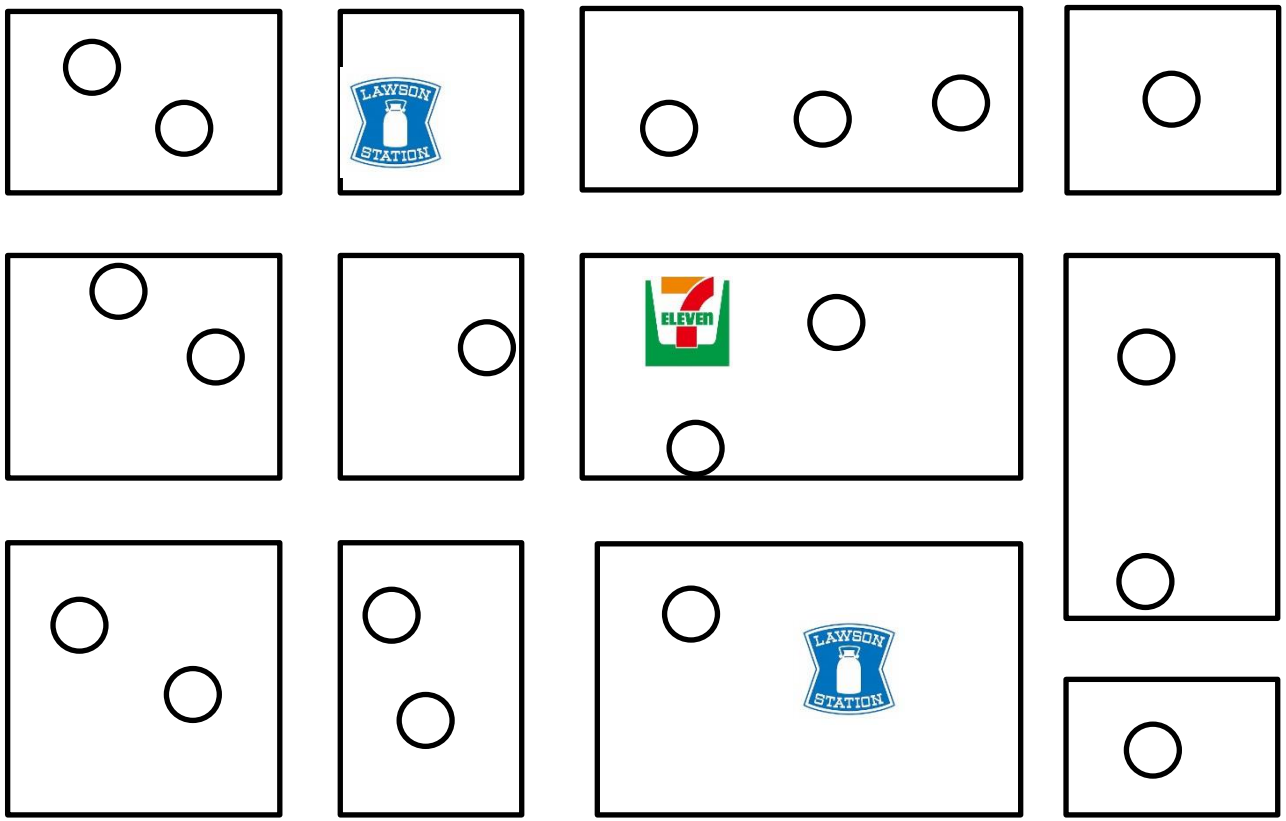
○：民家

：L店

：S店

ボロノイゲームとは？

コンビニのL店とS店が交互に出店する。



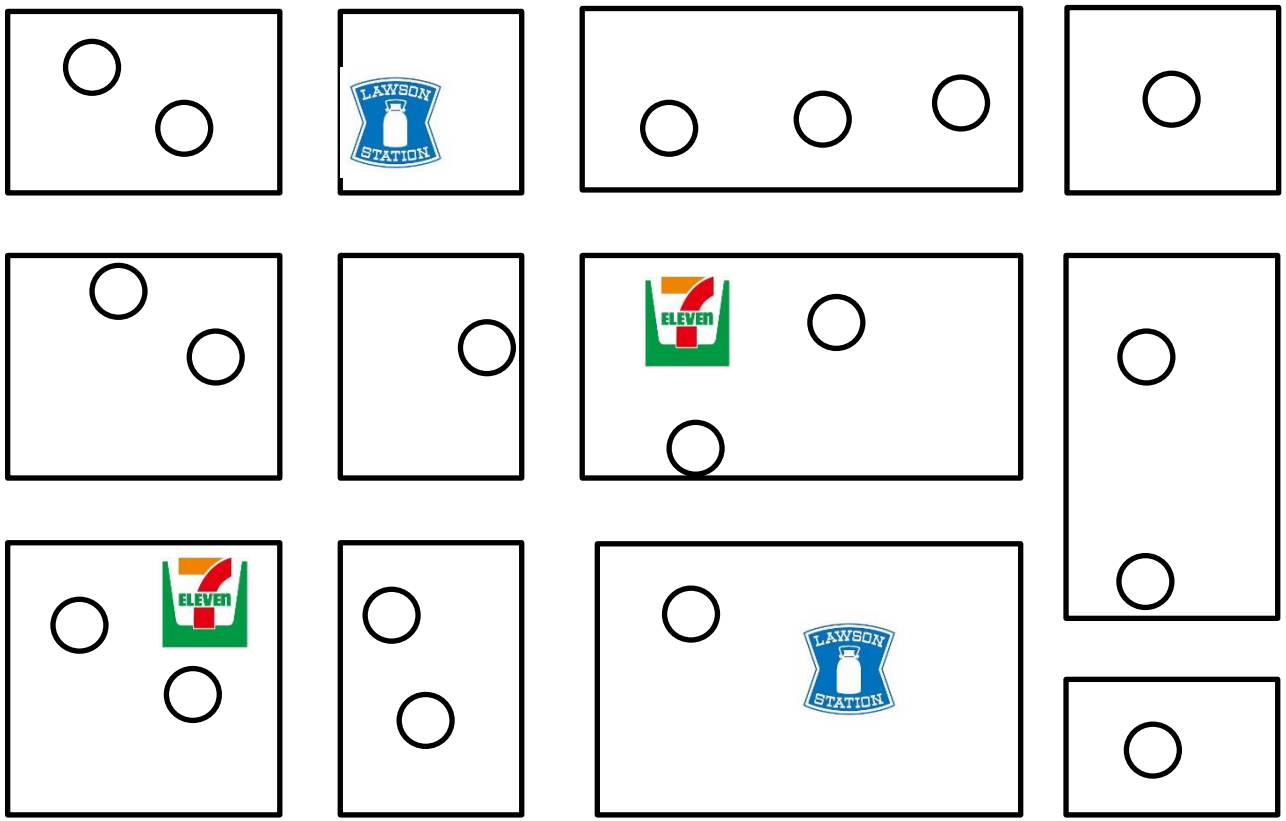
○ : 民家

: L店

: S店

ボロノイゲームとは？

コンビニのL店とS店が交互に出店する。



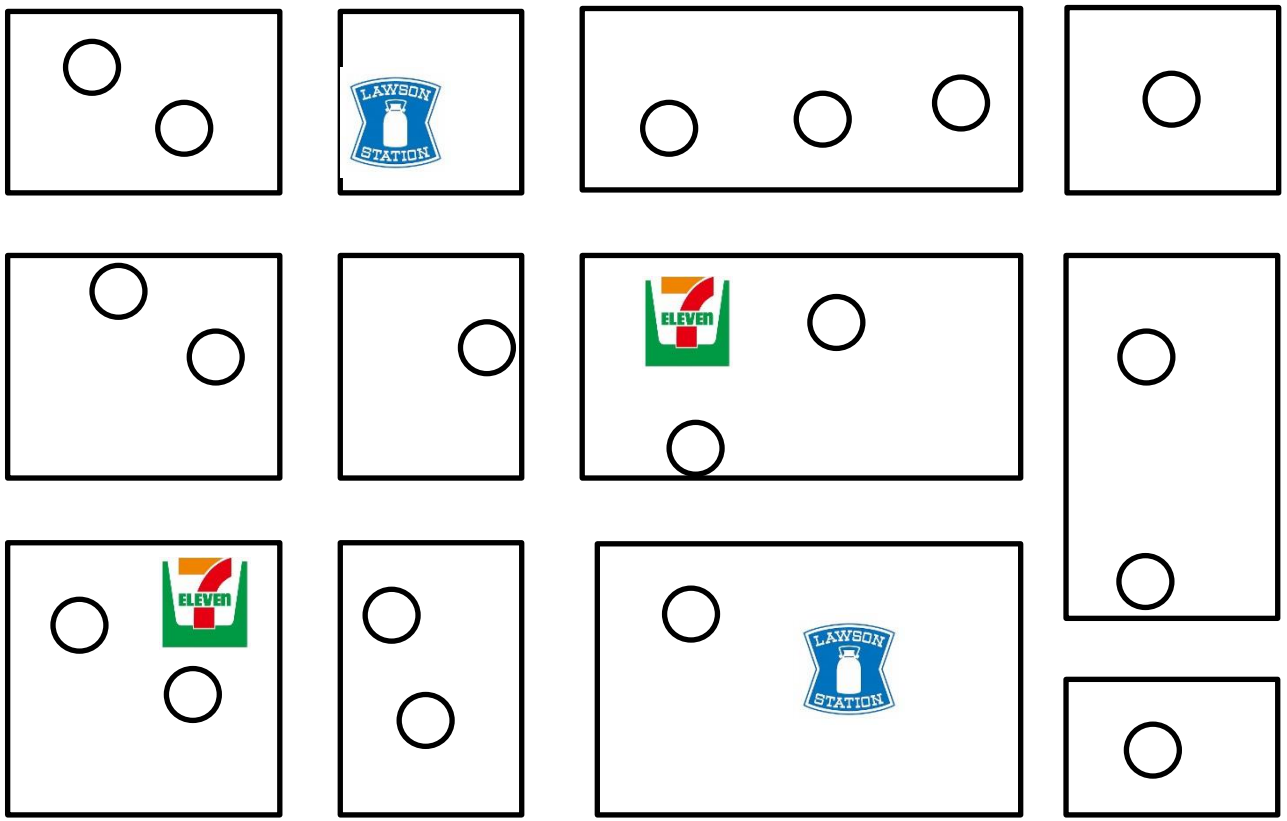
○ : 民家

: L店

: S店

ボロノイゲームとは？

より多くの客を呼ぶことを考えると･･･？



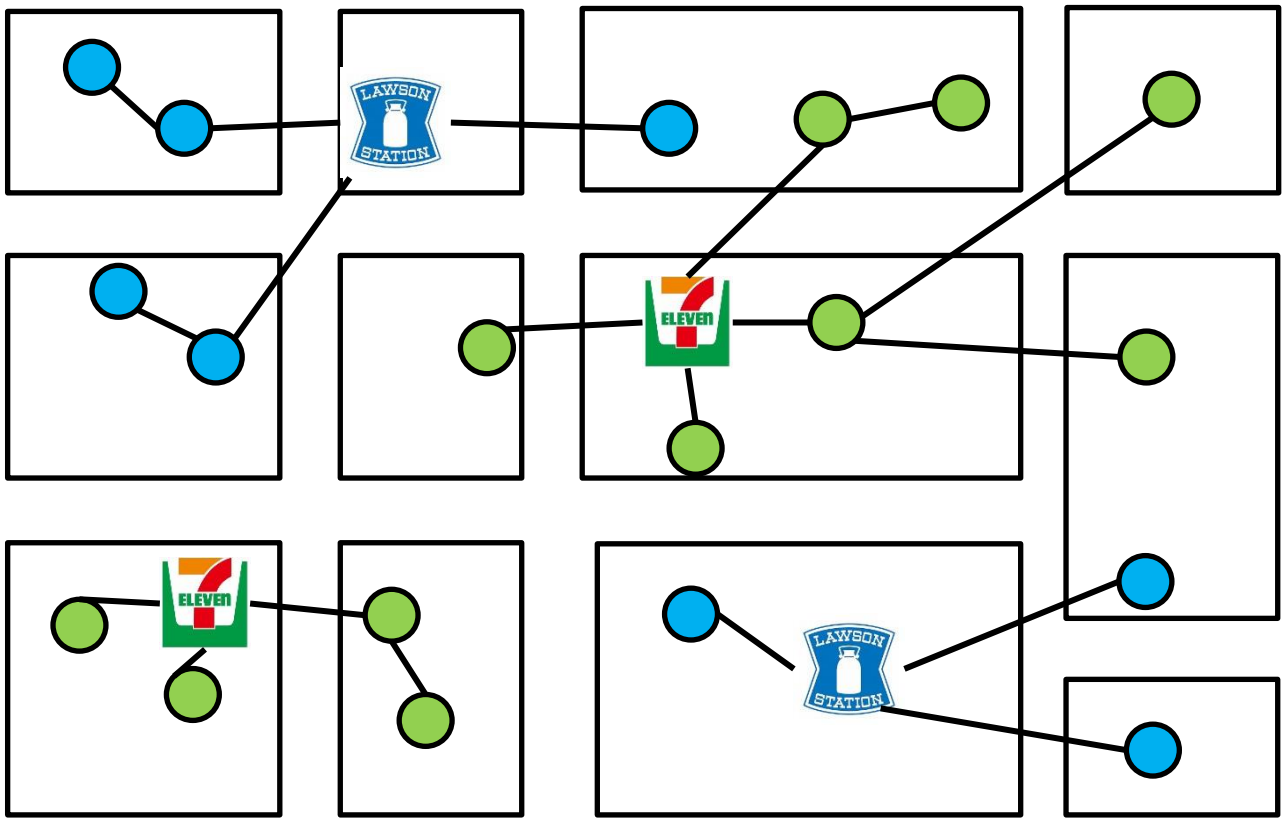
○：民家

：L店

：S店

ボロノイゲームとは？

距離の近い民家が多い方が有利！



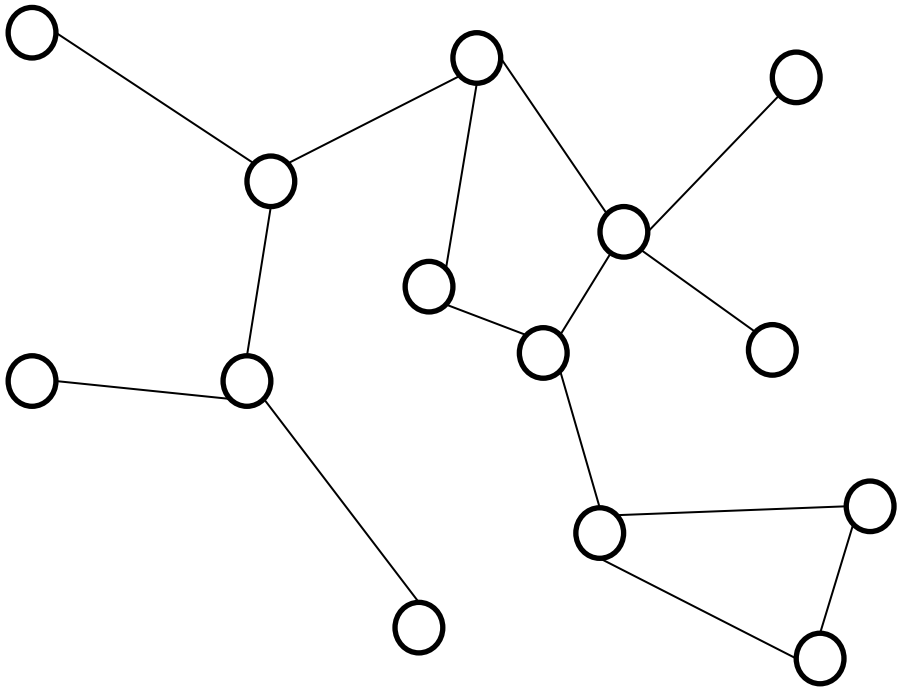
○：民家

：L店

：S店

(r,k)-ボロノイゲーム

グラフの上で対戦する1:1の陣取りゲーム



入力

無向グラフ

r:ラウンド数.

k:手数(＝店舗数)

(r,k)-ボロノイゲーム

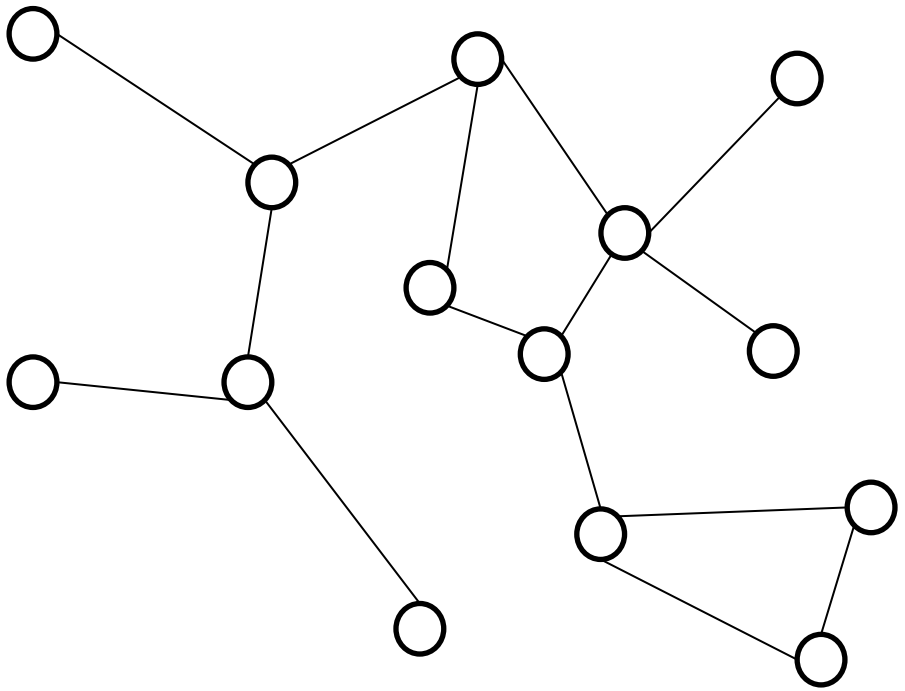
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

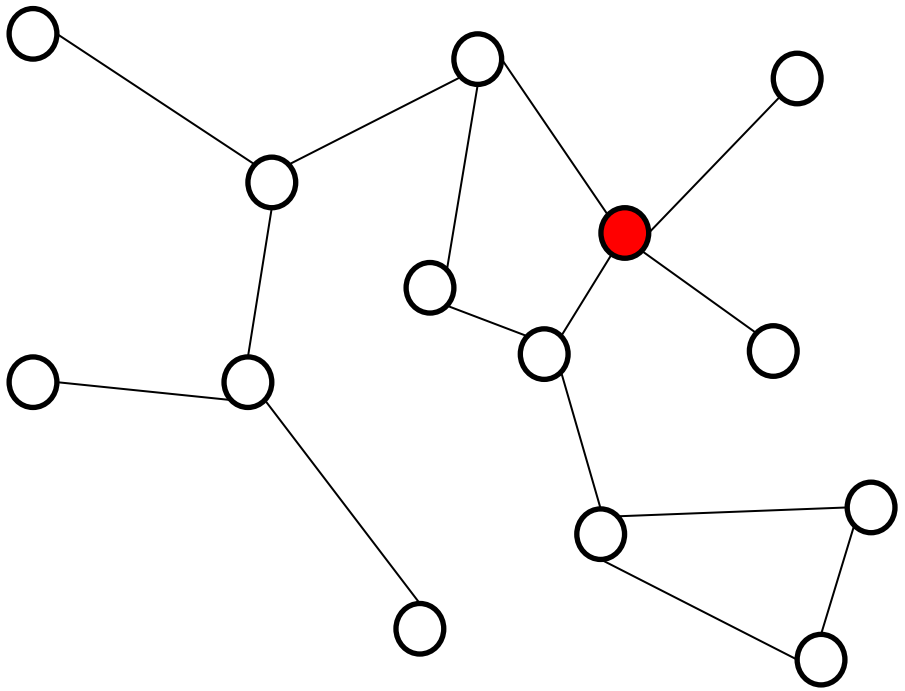
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

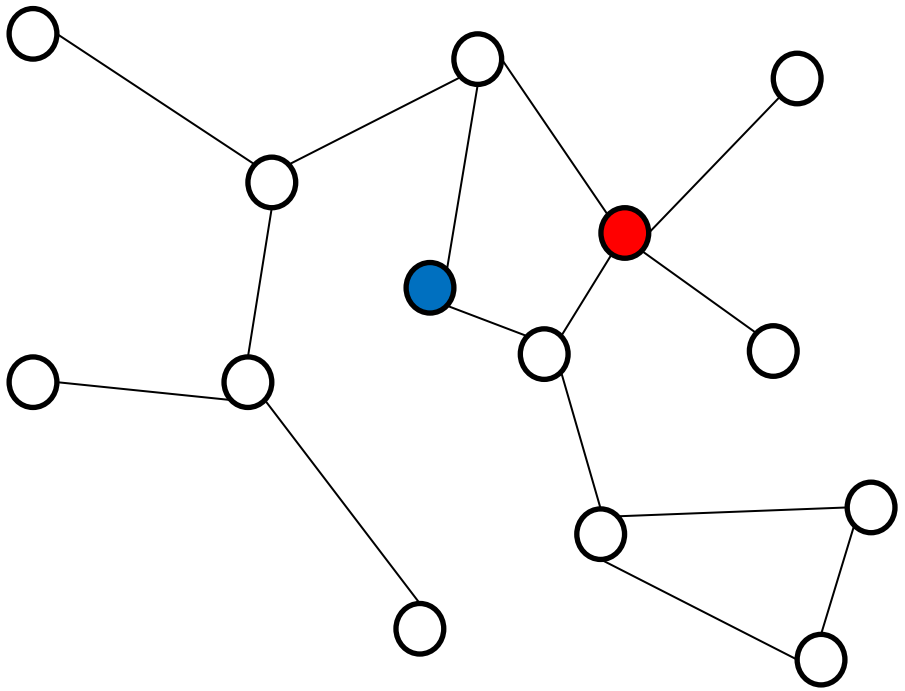
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

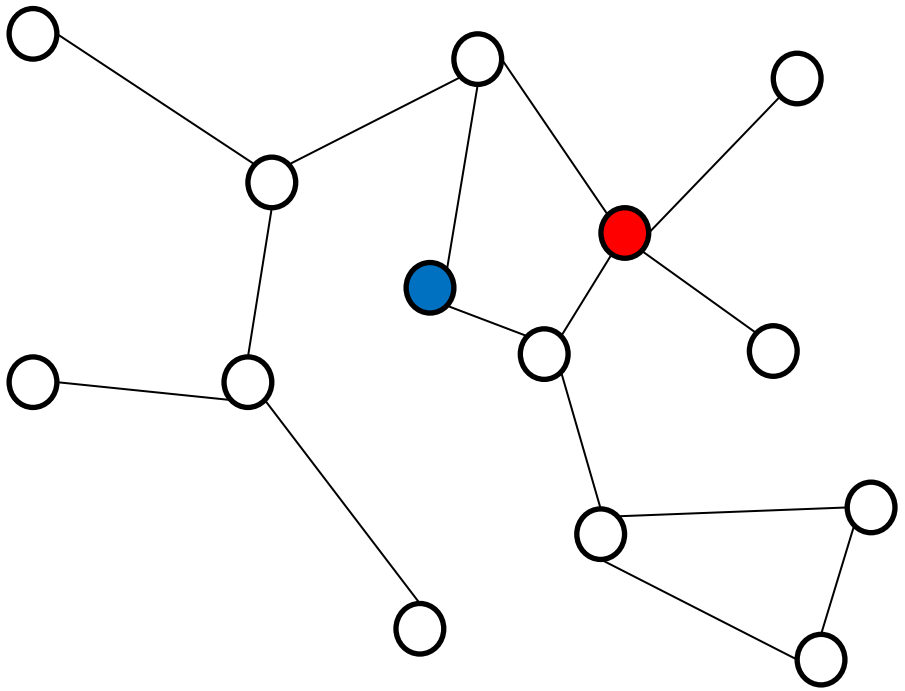
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

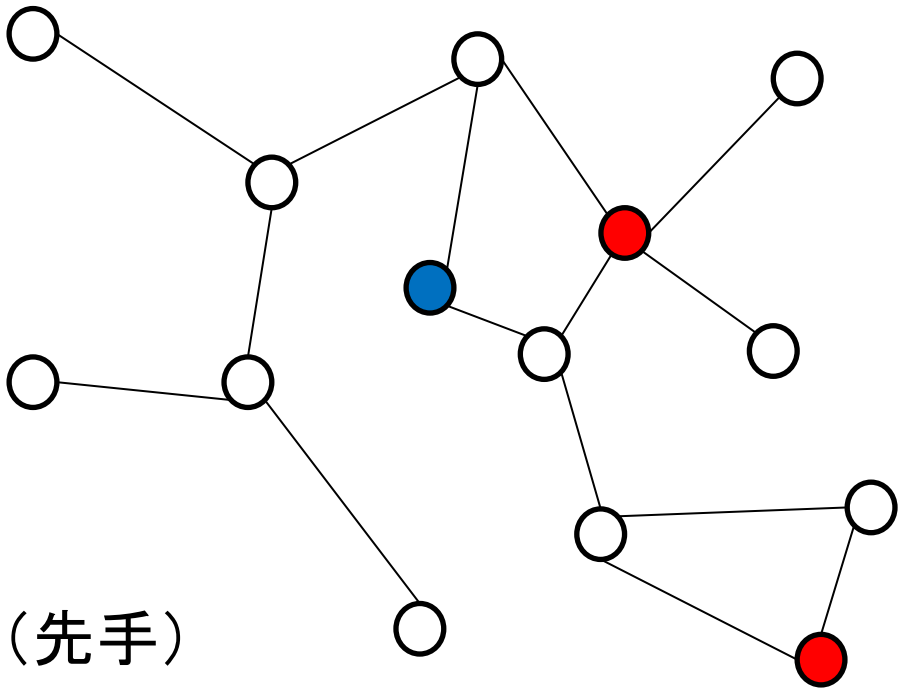
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

※2ラウンド目(先手)

(r,k)-ボロノイゲーム

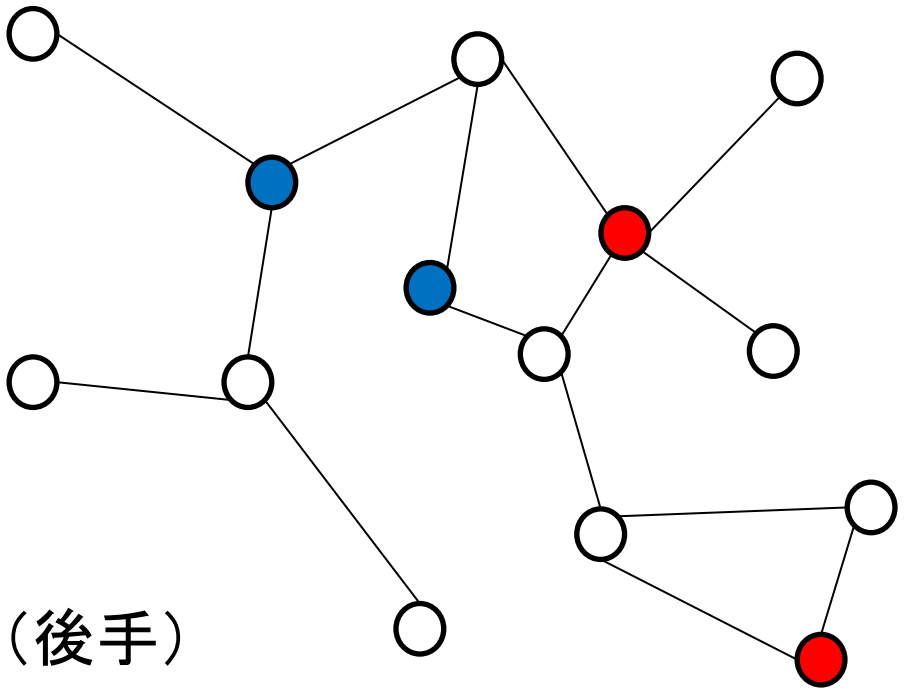
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

※2ラウンド目(後手)

(r,k)-ボロノイゲーム

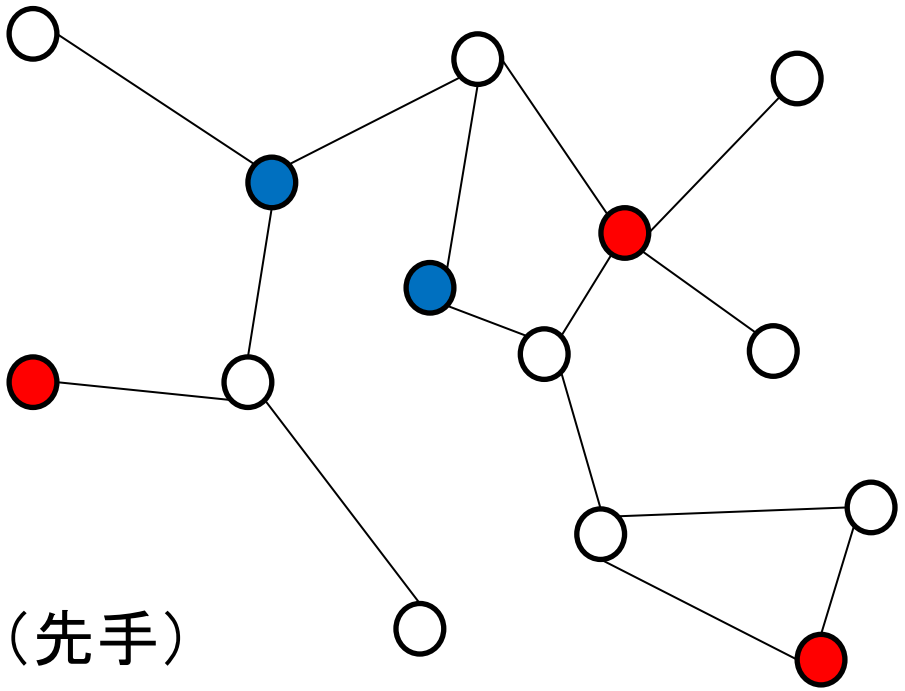
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

※3ラウンド目(先手)

(r,k)-ボロノイゲーム

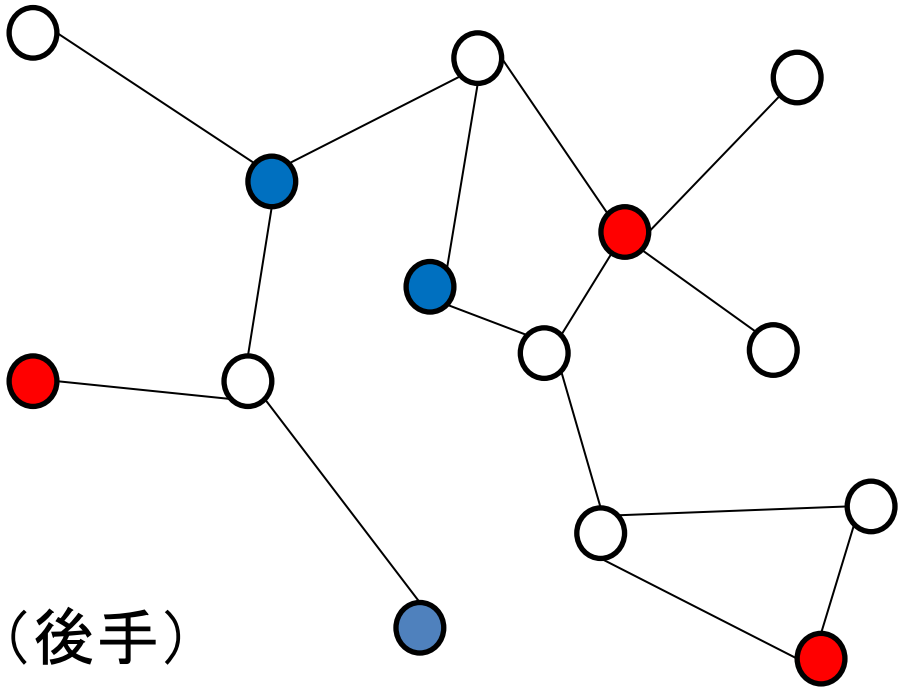
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

※3ラウンド目(後手)

(r,k)-ボロノイゲーム

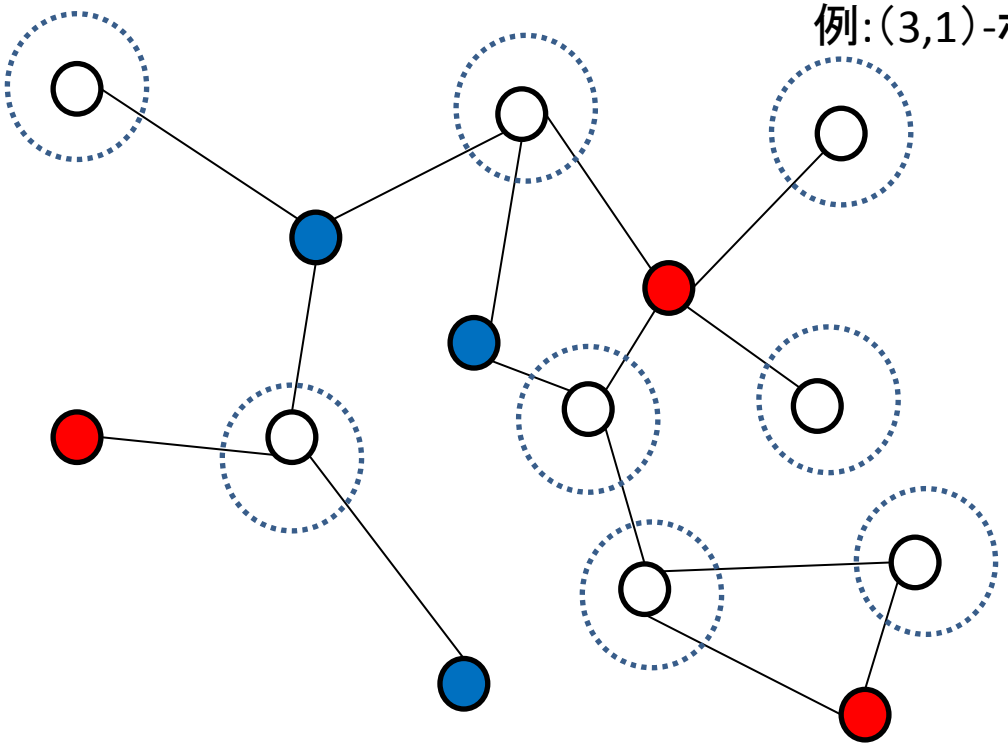
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

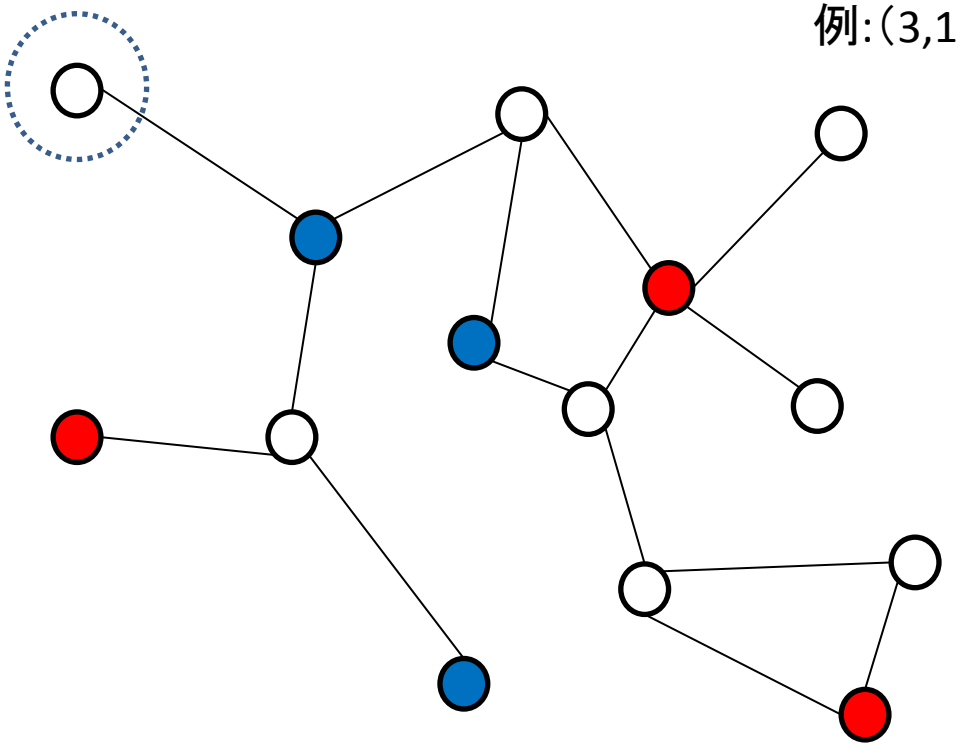
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

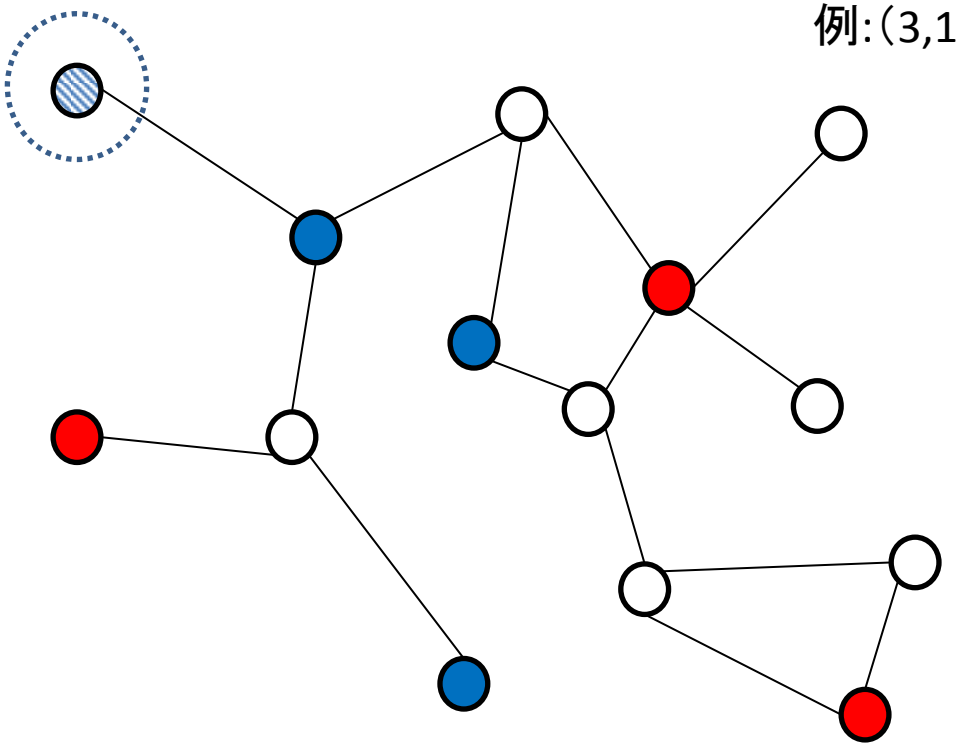
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

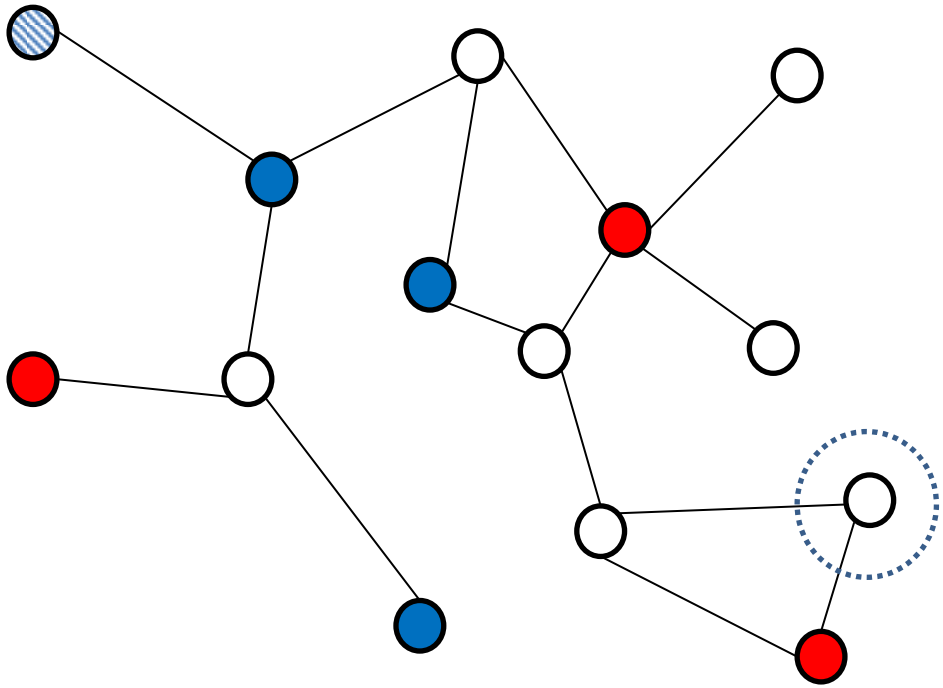
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

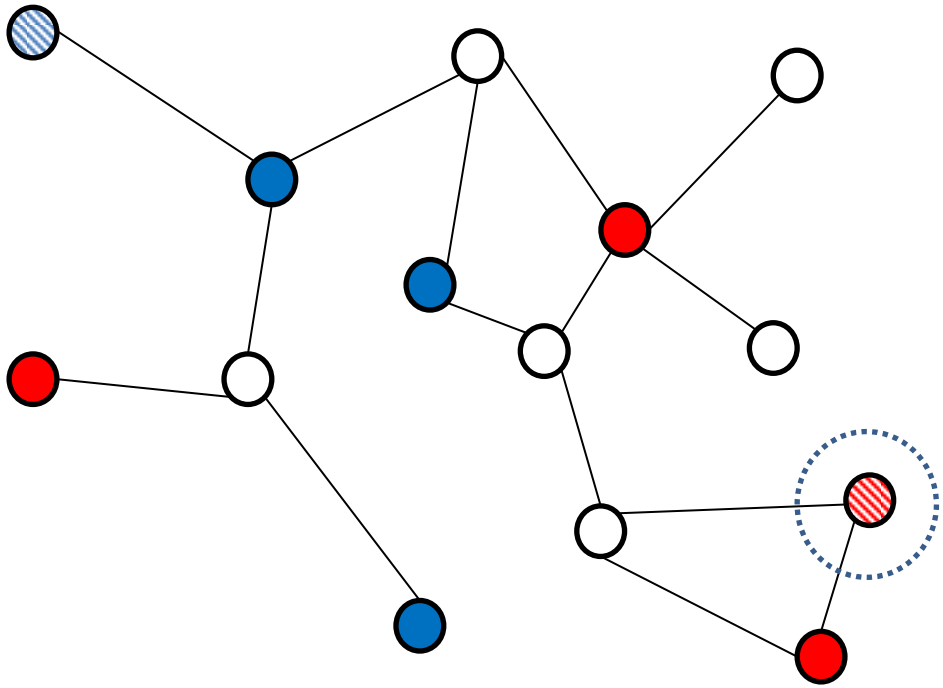
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

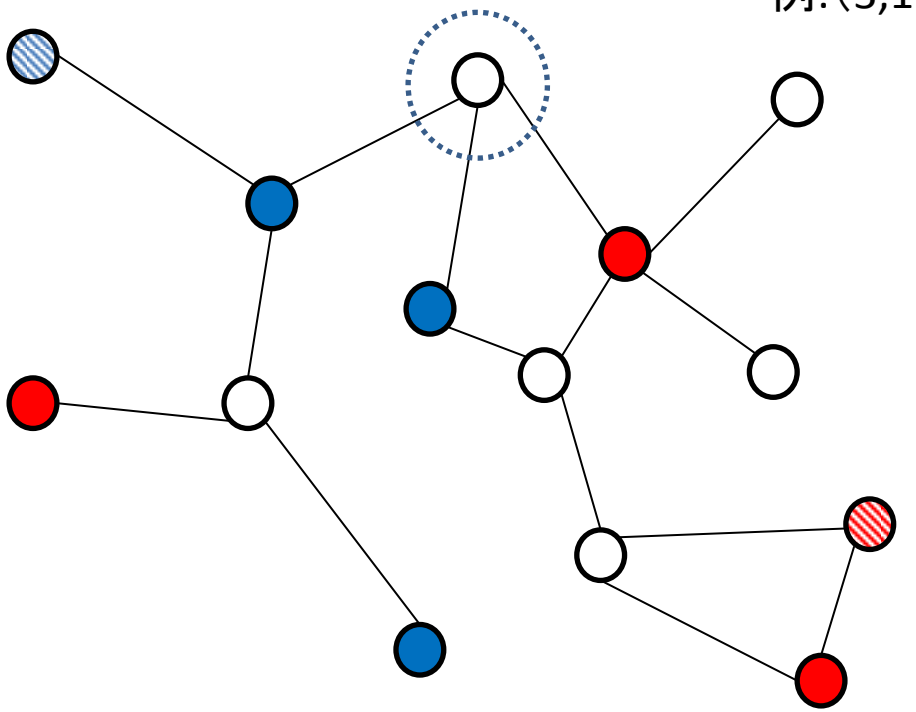
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

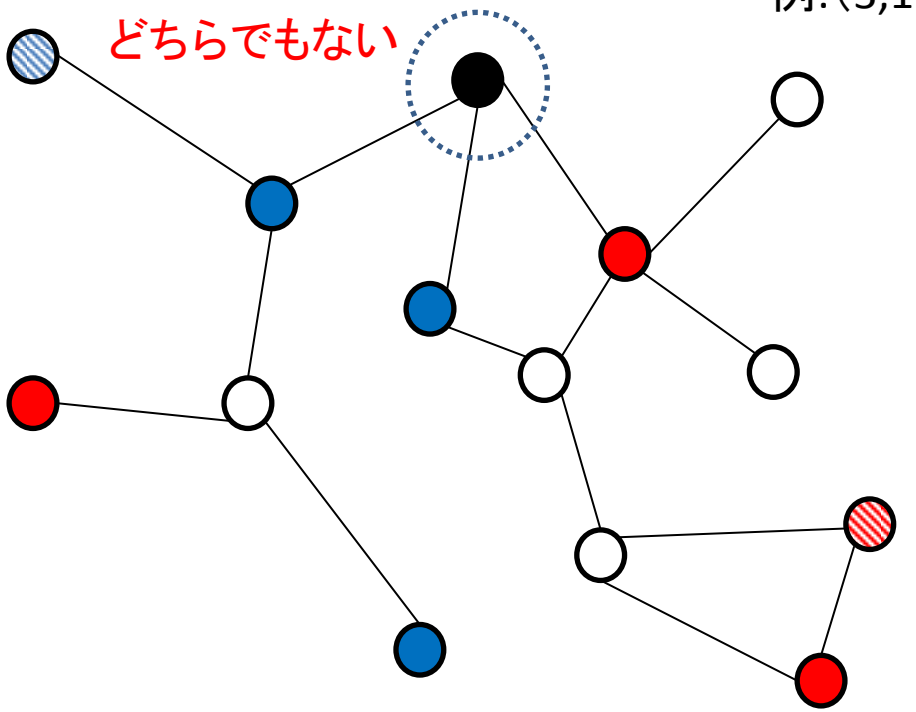
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

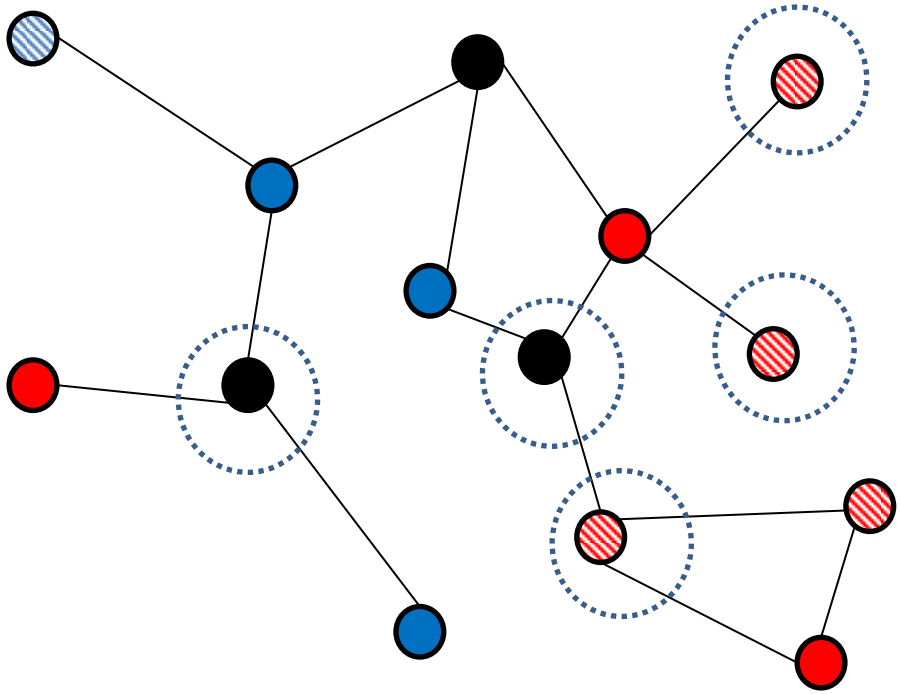
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



先手

後手

(r,k)-ボロノイゲーム

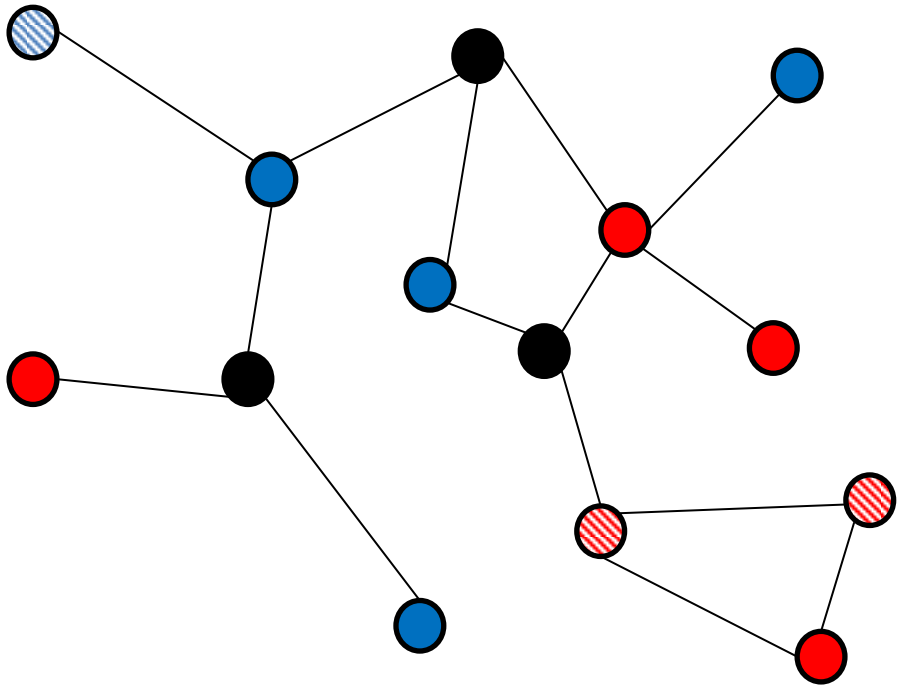
1.先手と後手が交互にk個ずつ陣地を選ぶ

2.これをラウンド数r回だけ繰り返す

3.残りの頂点については、最も距離が近い方の陣地とする

4.陣地の数が多い方が勝ち

例:(3,1)-ボロノイゲーム



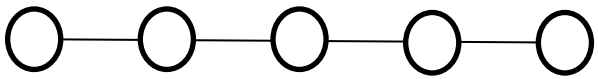
先手

後手

既存研究

・パスの(r,1)-ボロノイゲーム

※互いが最善手を打つと仮定



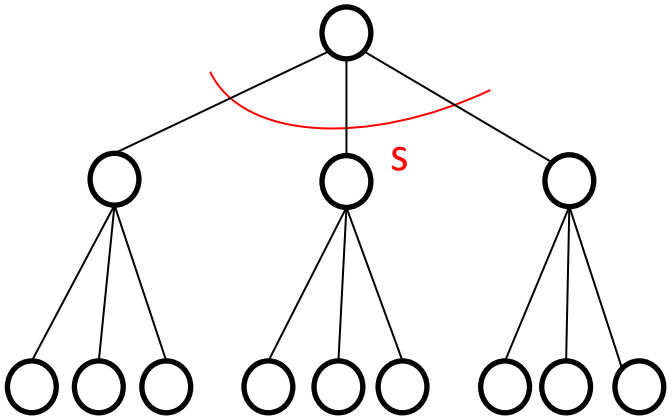
・r=1かつ頂点が奇数個なら先手勝利

・偶数個なら引き分け

[清見ら, 2010年]

・完全s分木の(1,k)-ボロノイゲーム

完全s分木：全ての内部頂点がちょうどs個の子を持ち、全ての葉が同じ高さの根つき木



・s<2kなら先手勝利

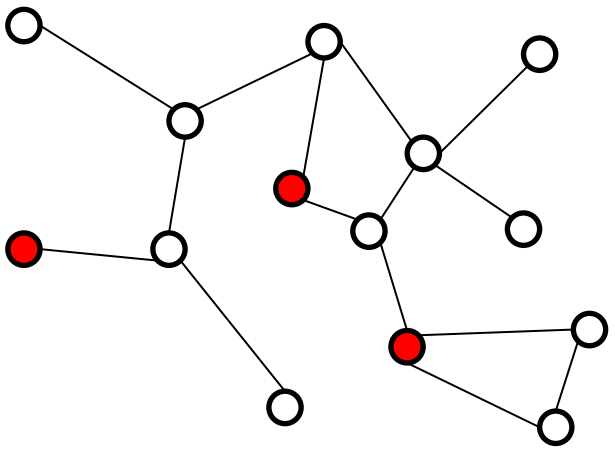
s≧2kのとき

・s分木が十分大きければ先手勝利

・それ以外は引き分け [寺本ら, 2006年]

既存研究

・一般のグラフの(1,k)-ボロノイゲーム



先手のk個の頂点が与えられたとき、後手が勝てるかどうかの判定はNP完全

[寺本ら, 2006年]

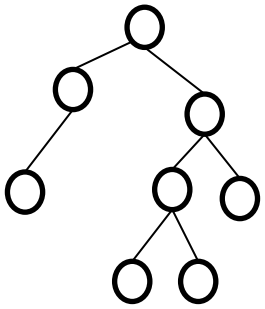
今回の発表内容

木の(1,k)-ボロノイゲームにおいて、先手のk個の頂点が与えられたとき、(後手)−(先手)の値の最大値を求めるアルゴリズム

・二分木

各頂点の子の数が高々2

頂点数n

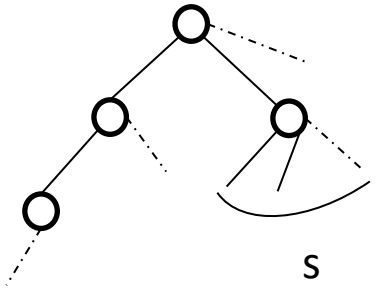


計算時間 $\mathcal{O}(k^2n^4)$

・s分木

各頂点の子の数が高々s

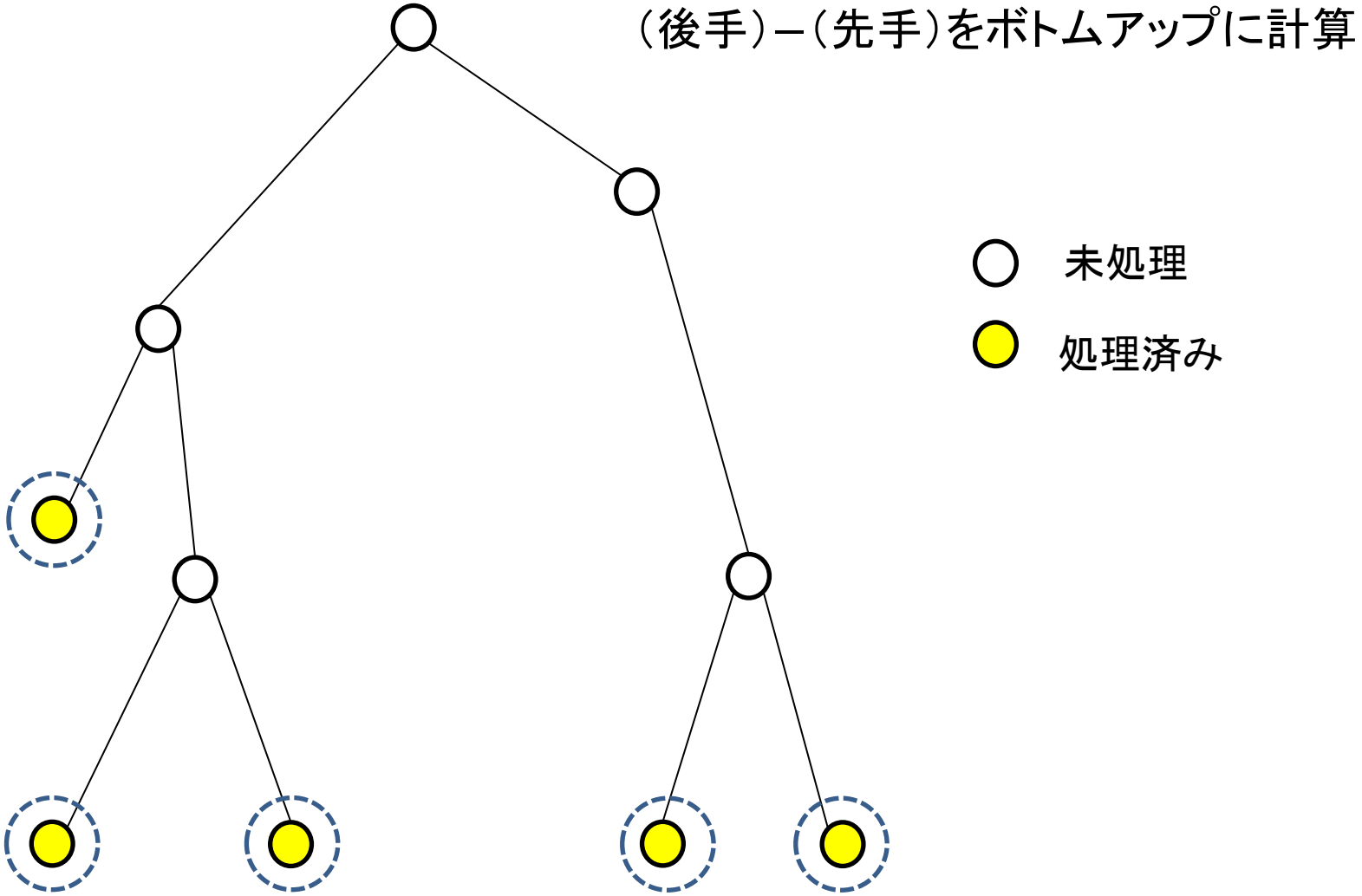
頂点数n



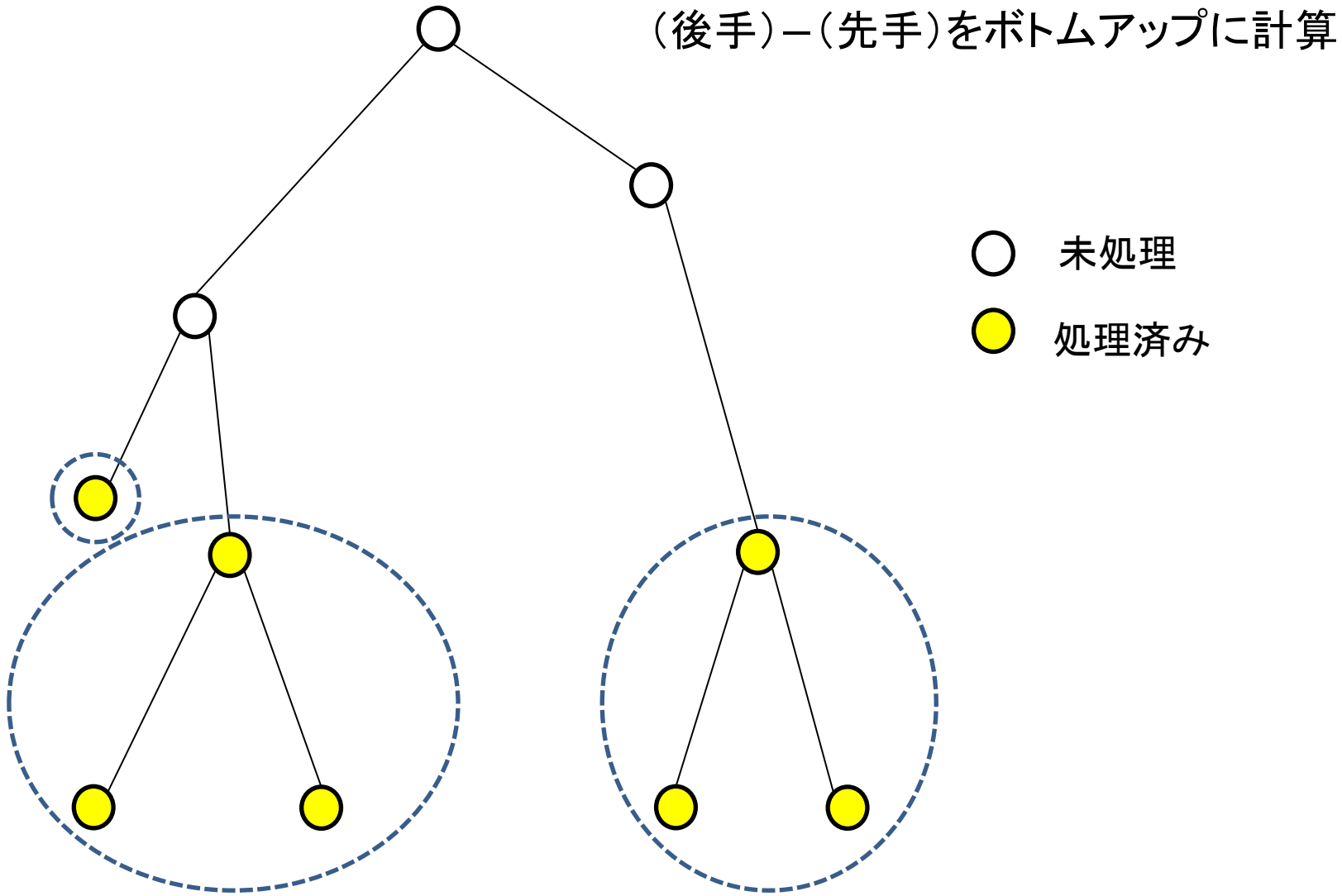
計算時間 $\mathcal{O}\left(\frac{(k+s-1)!}{k!\,(s-1!)}ksn^{s+2}\right)$

※アルゴリズムには動的計画法を用いる

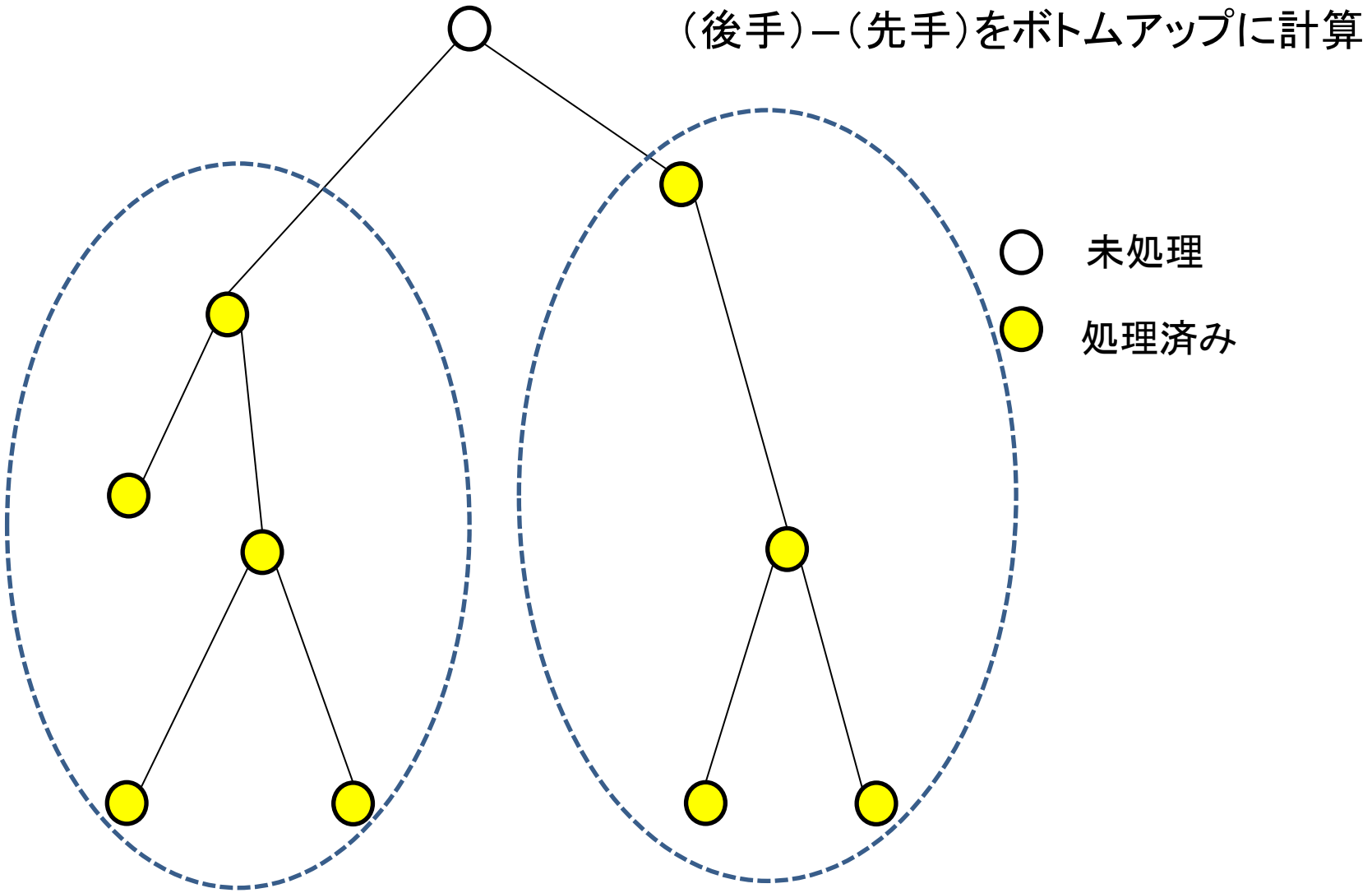
アルゴリズムのイメージ



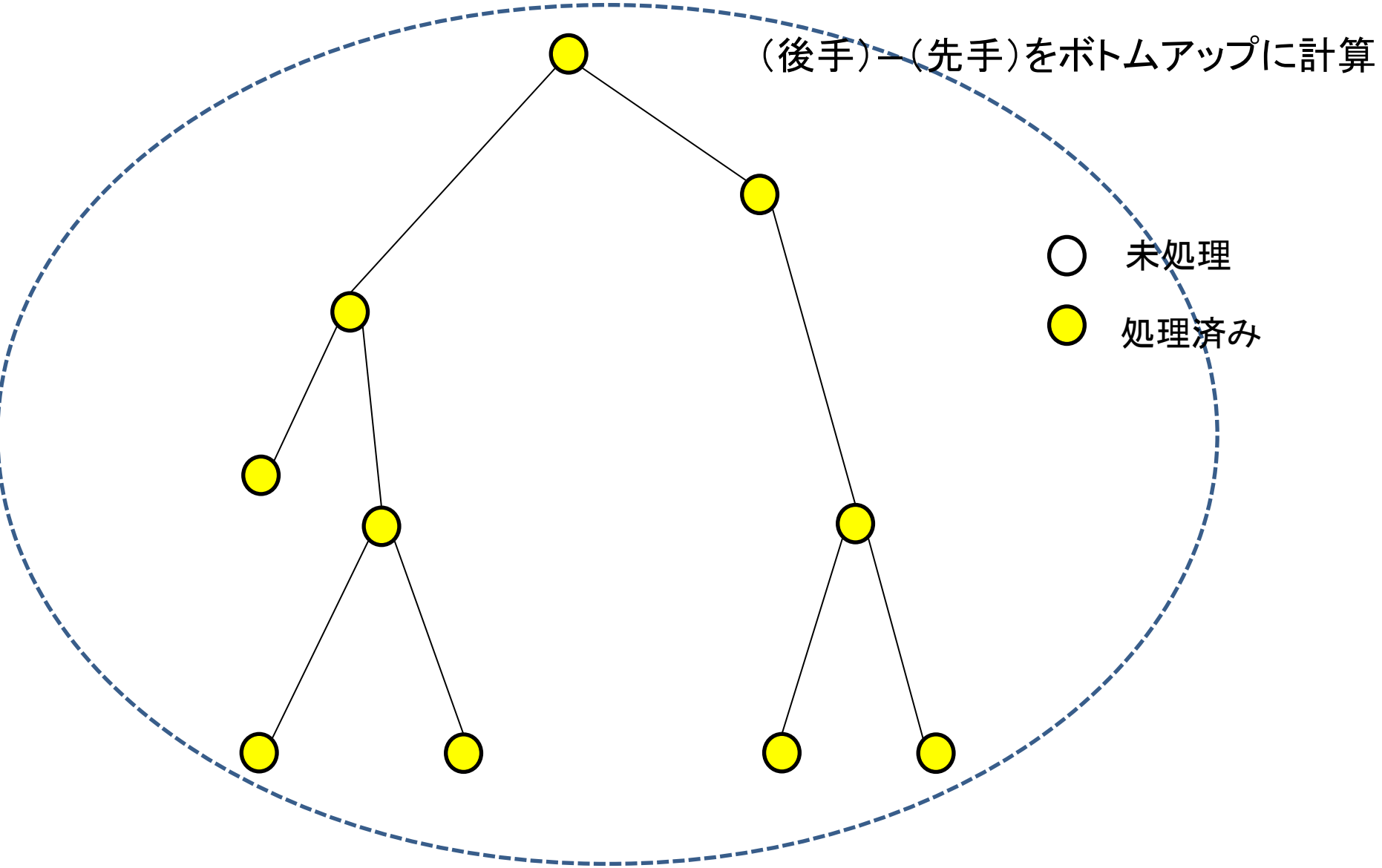
アルゴリズムのイメージ



アルゴリズムのイメージ

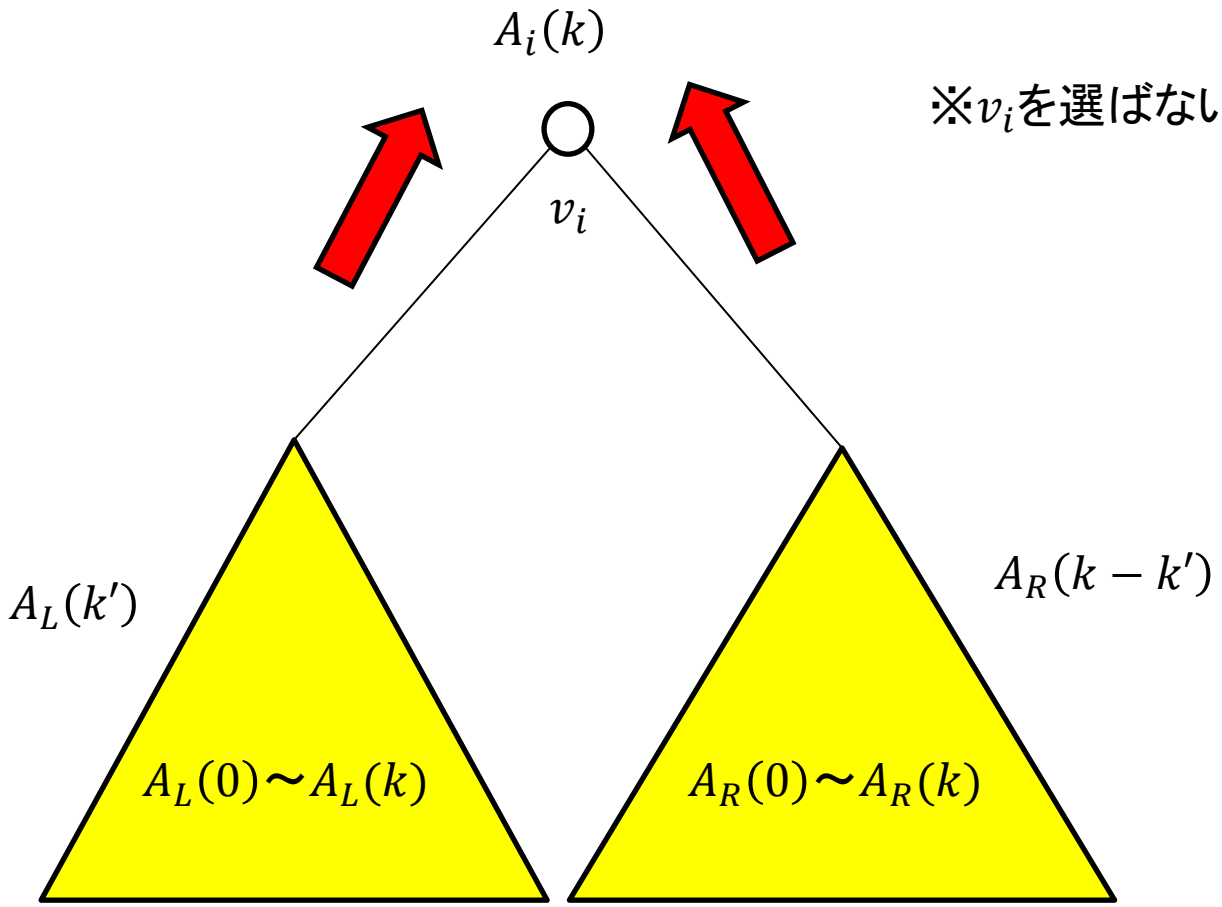


アルゴリズムのイメージ



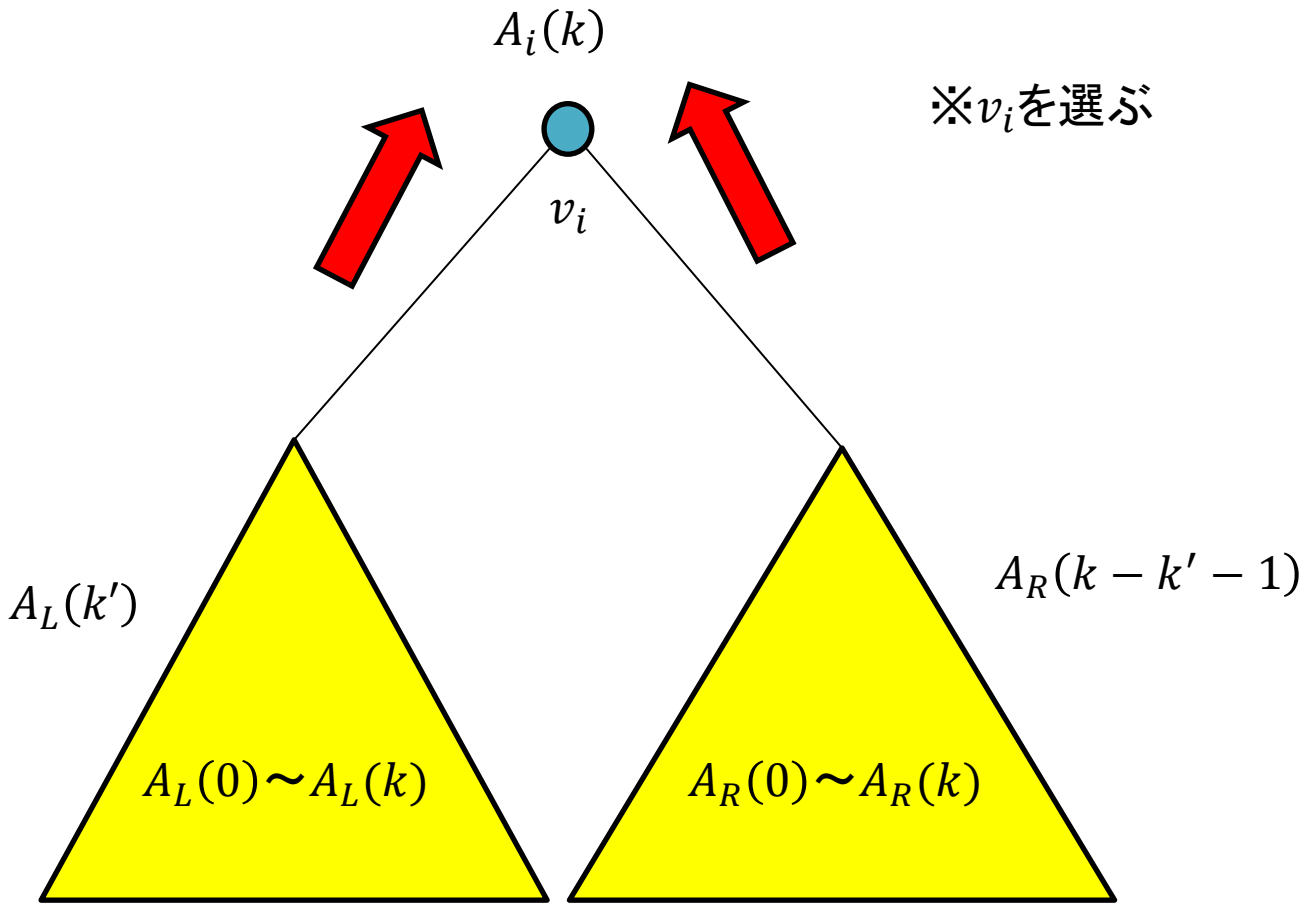
基本的なアイディア

$A_i(k)$: v_iを根とする部分木内でk個選ぶときの（後手）−（先手）の最大値

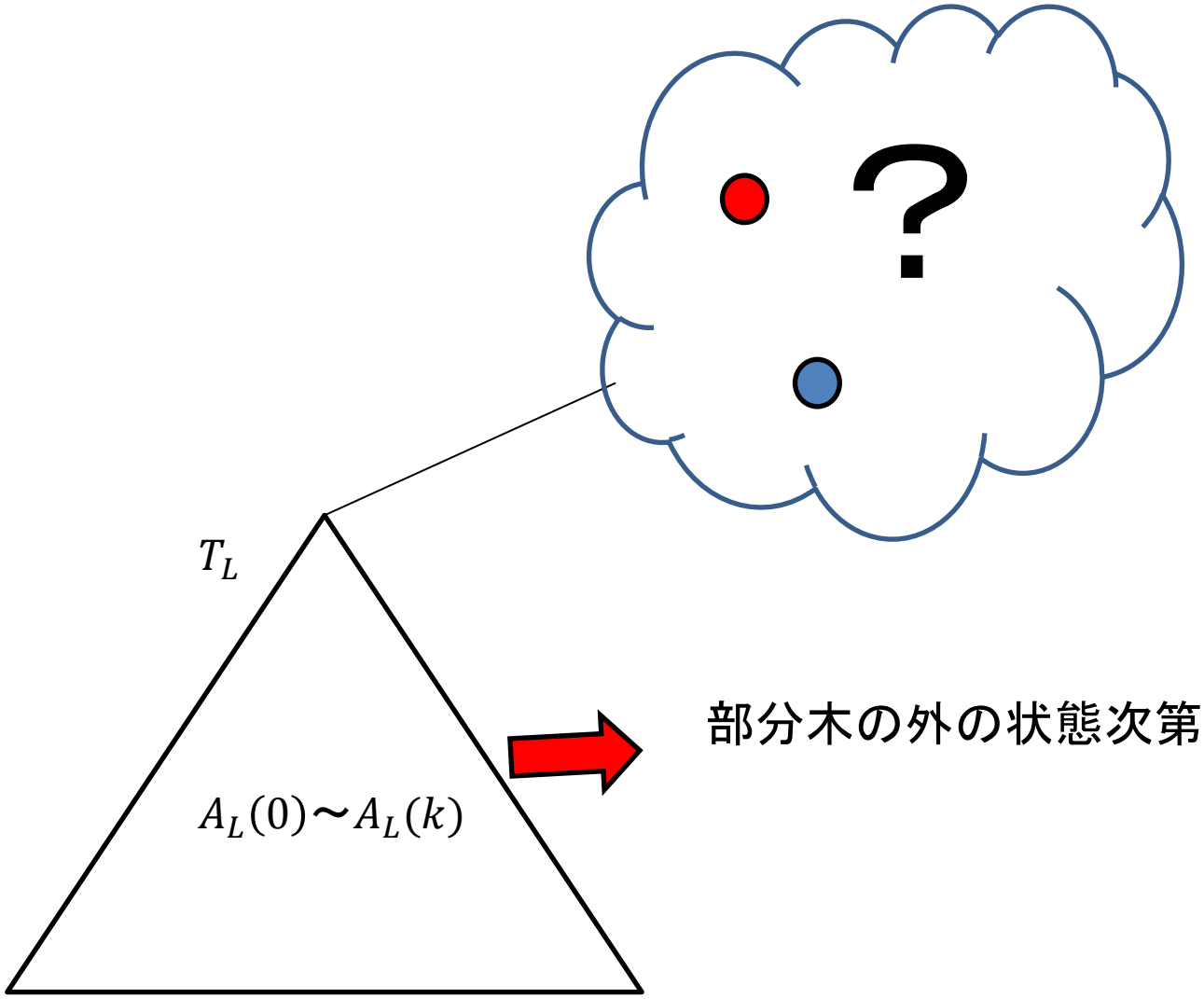


基本的なアイディア

$A_i(k)$: v_iを根とする部分木内でk個選ぶときの（後手）−（先手）の最大値

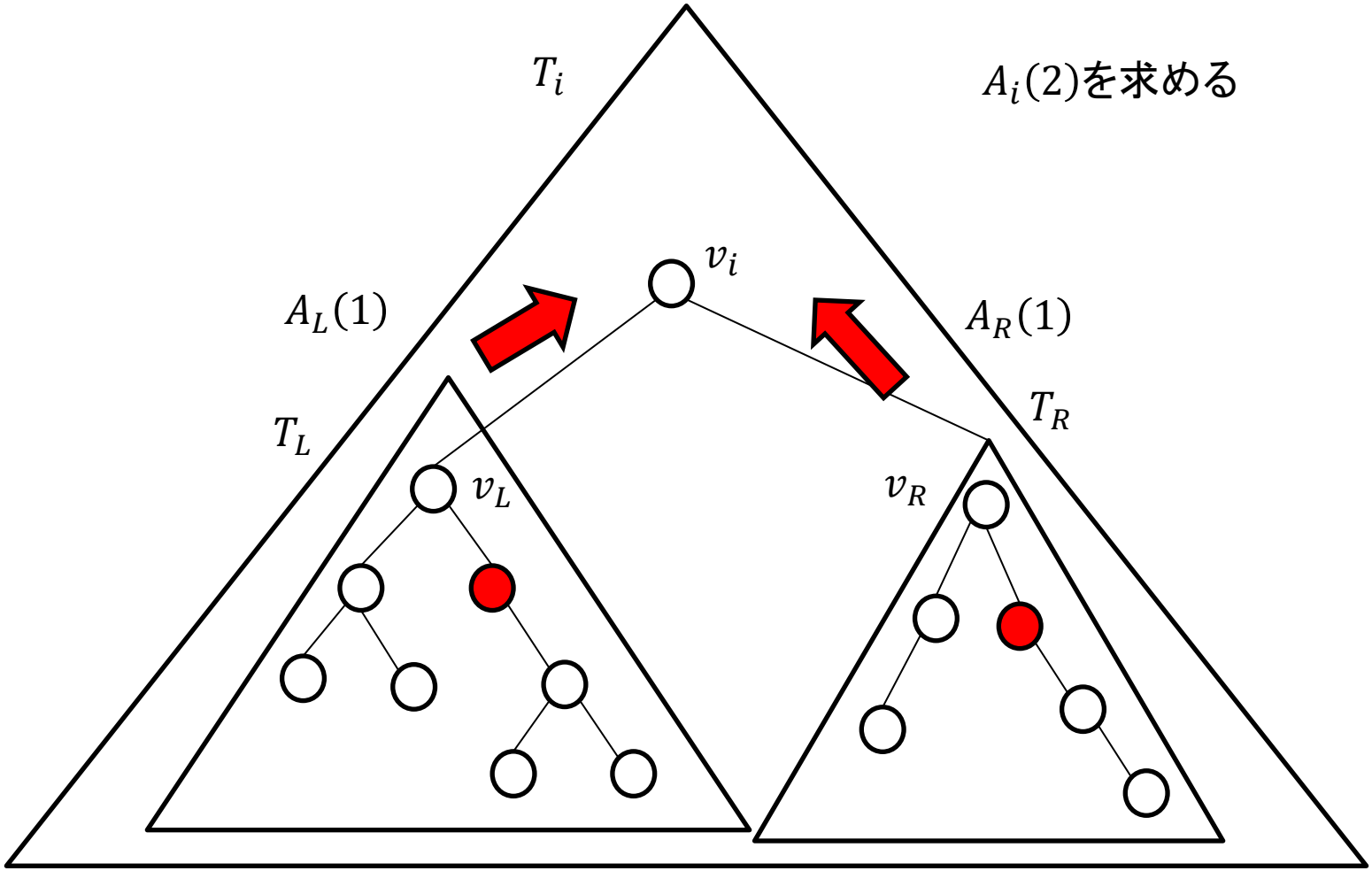


だがしかし上手くいかない

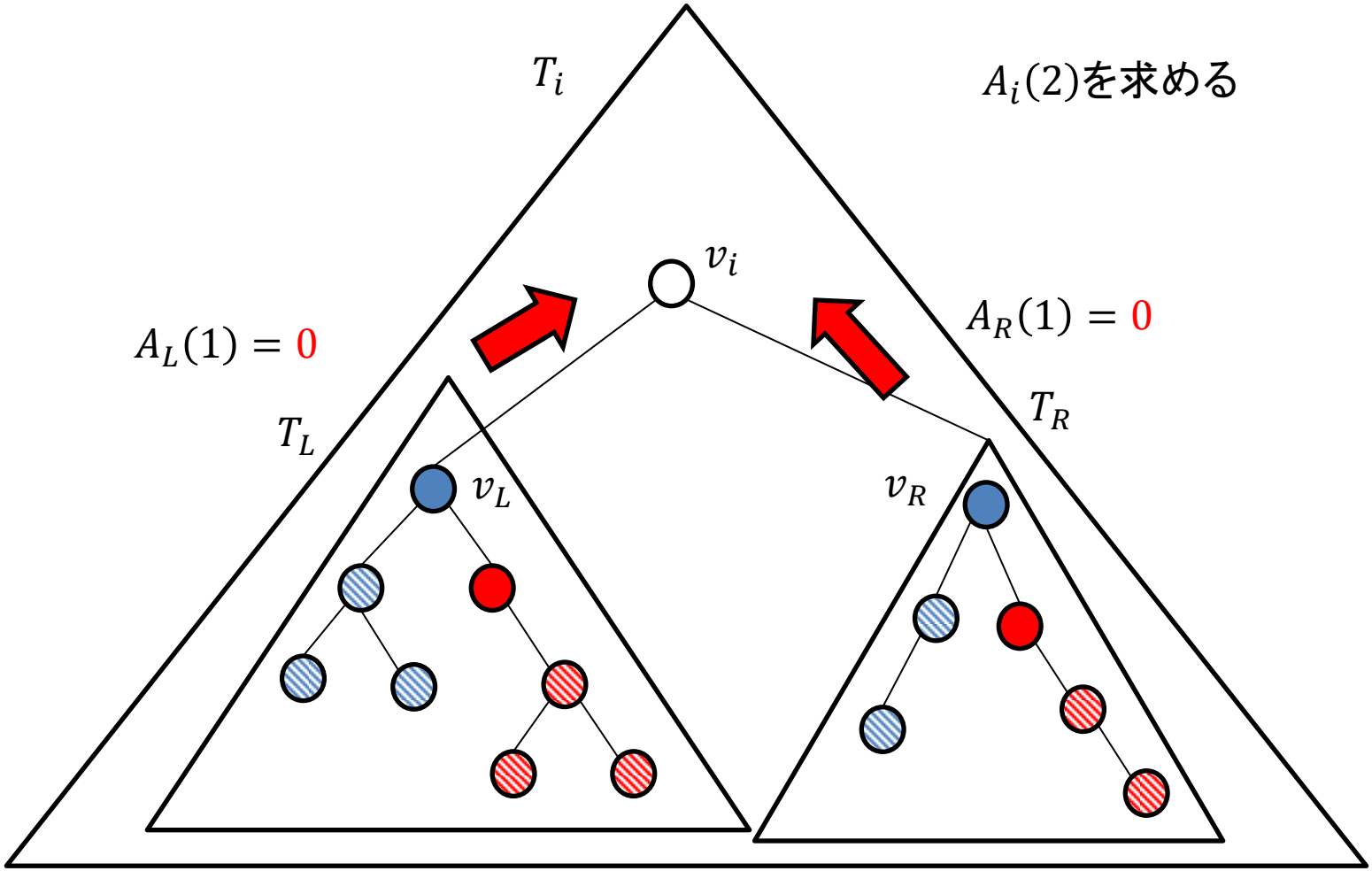


部分木の外の状態次第で値が変わってしまう

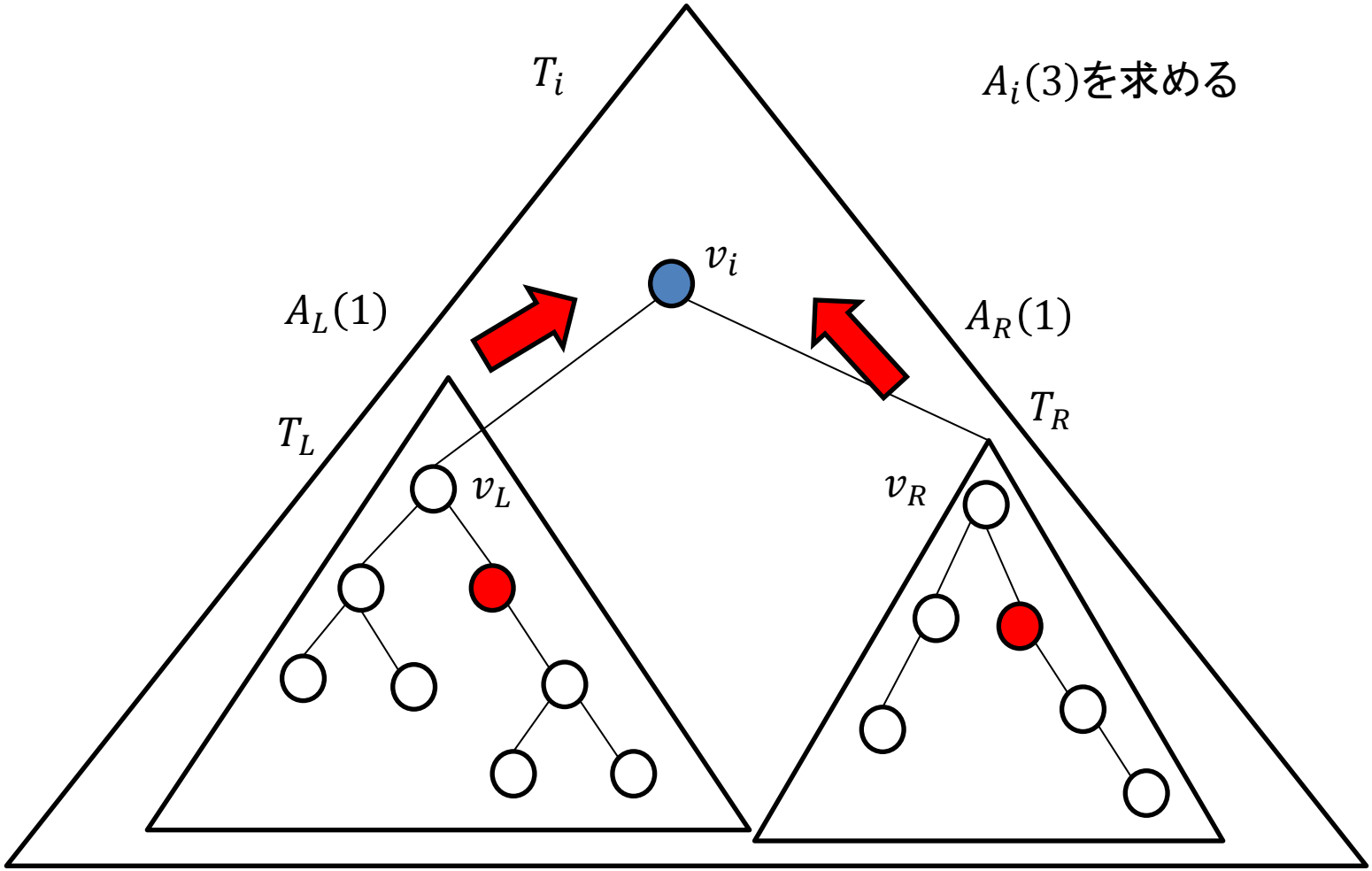
だがしかし上手くいかない



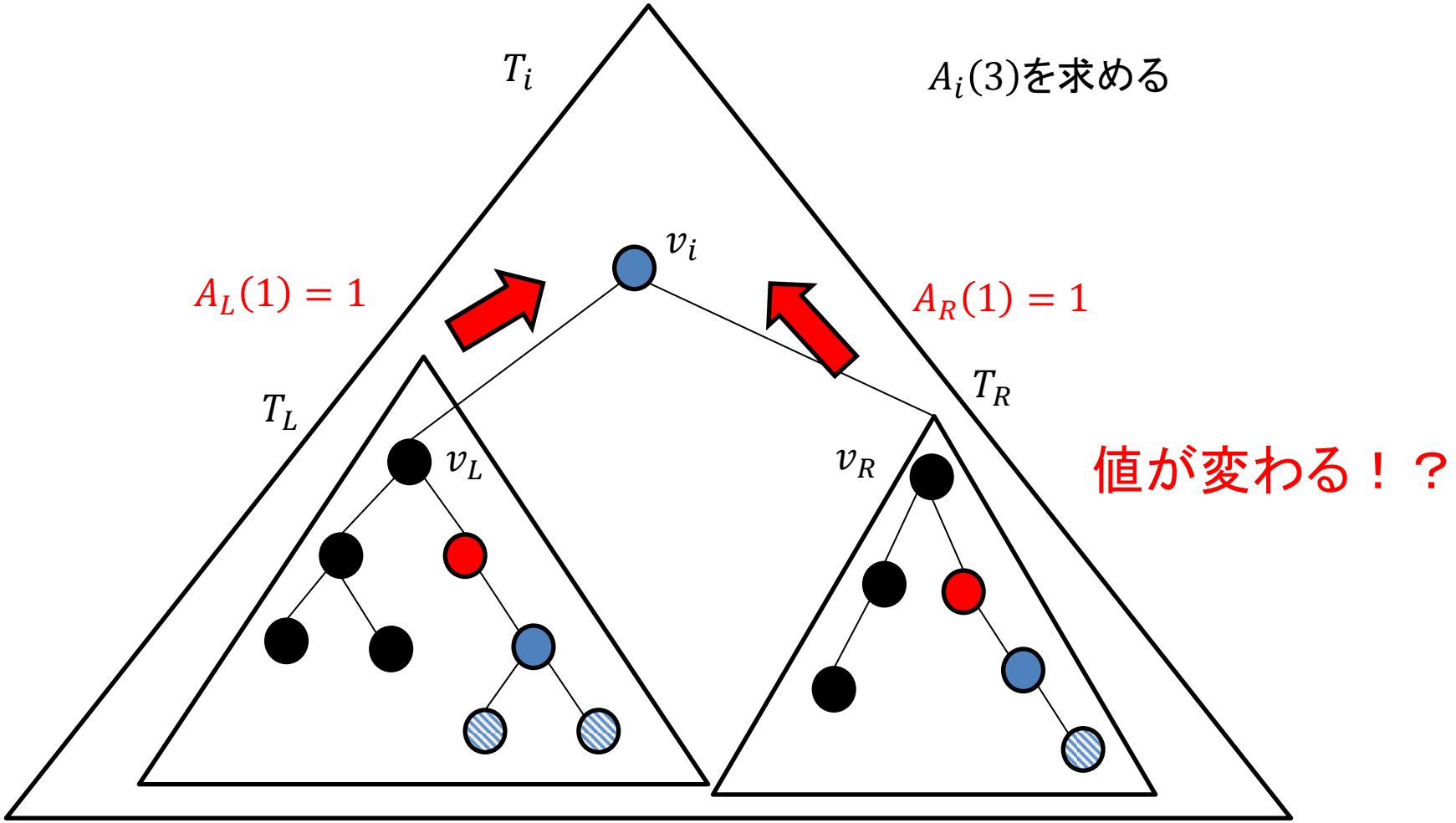
だがしかし上手くいかない



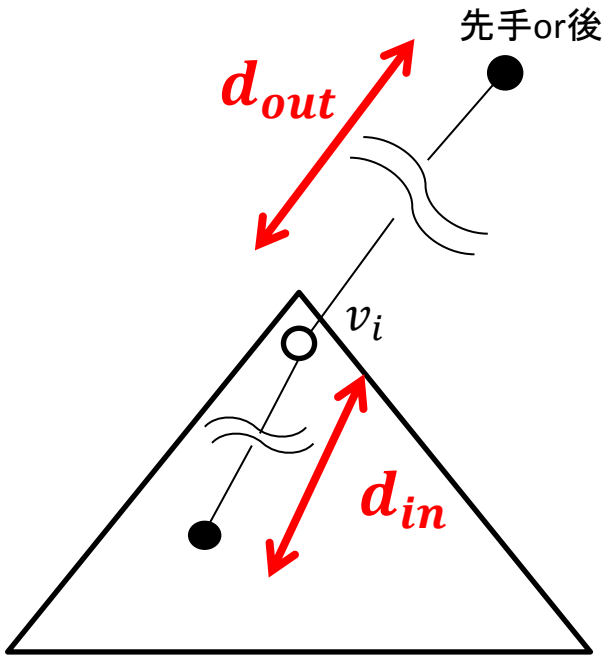
だがしかし上手くいかない



だがしかし上手くいかない



ならばどうする？



$A_i(k)$に以下の情報を持たせればよい

外部距離: d_{out}

v_iの子孫以外の頂点のうち、最も近い先手または後手の頂点とv_iとの距離

内部距離: d_{in}

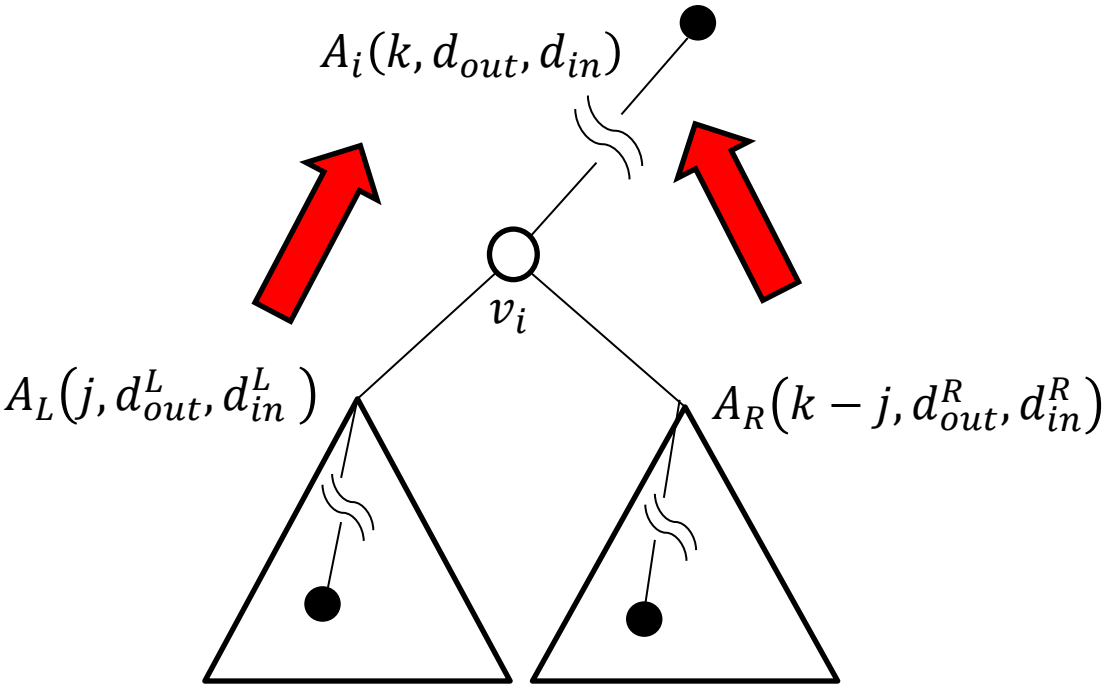
v_iの子孫のうち、最も近い先手または後手の頂点とv_iとの距離

	先1	..	後1	...
$A_L(k)$				

二分木の場合

※v_iを選ばない

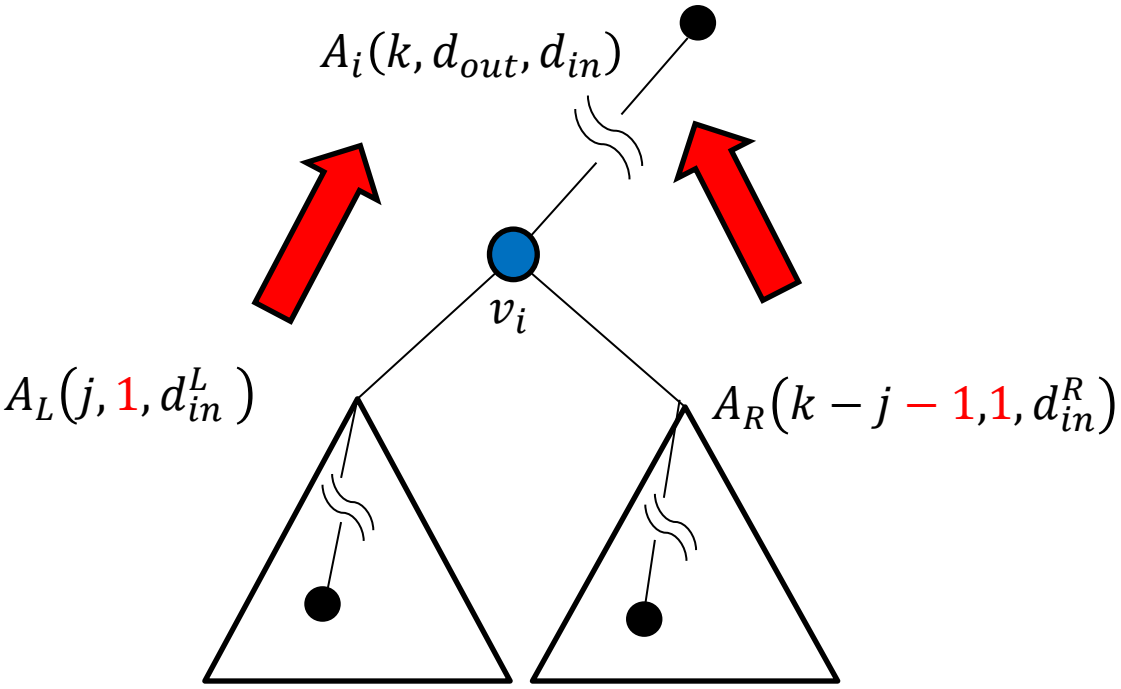
$A_{根}(k, \infty, \cdot)$を求めて終了



二分木の場合

※v_iを選ぶ

$A_{根}(k, \infty, \cdot)$を求めて終了

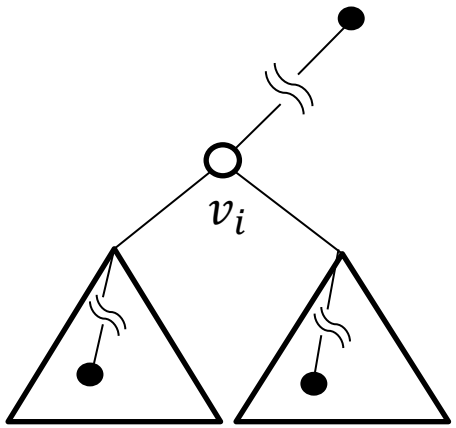


二分木の場合

$A_i(k, d_{out}, d_{in})$

※v_iを選ばないとき

$$\max \begin{cases} \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^R \le n}} \{ A_L(j, d_{out}+1, d_{in}-1) + A_R\left(k-j, d_{out}+1, d_{in}^R\right)\} \\ \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^L \le n}} \{A_L(j, d_{out}+1, d_{in}^L) + A_R(k-j, d_{out}+1, d_{in}-1)\} \end{cases}$$

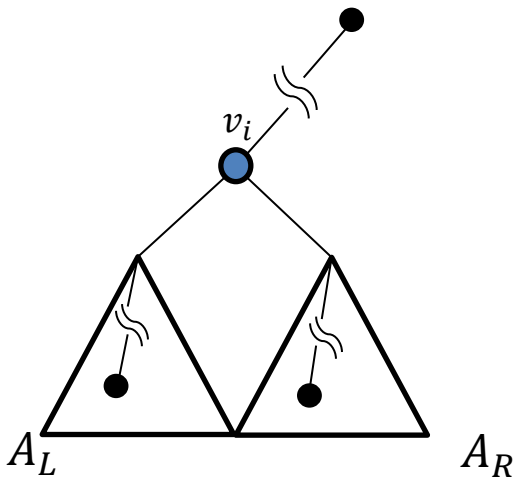


二分木の場合

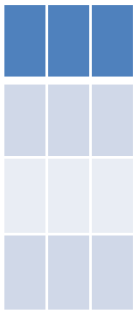
$A_i(k, d_{out}, d_{in})$

※v_iを選ぶとき

$$\max \begin{cases} \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^R \le n}} \{ A_L(j, 1, d_{in}-1) + A_R(k-j-1, 1, d_{in}^R) \} \\ \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^L \le n}} \{ A_L(j, 1, d_{in}^L) + A_R(k-j-1, 1, d_{in}-1) \} \end{cases}$$



二分木の場合

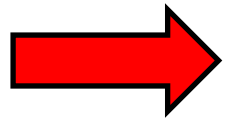


$A_i(k, d_{out}, d_{in})$の表のサイズは高々kn^2

jのループの回数がk

d_{in}^L, d_{in}^Rのループがそれぞれ高々n

n個の頂点に対してA_iを求める



計算時間

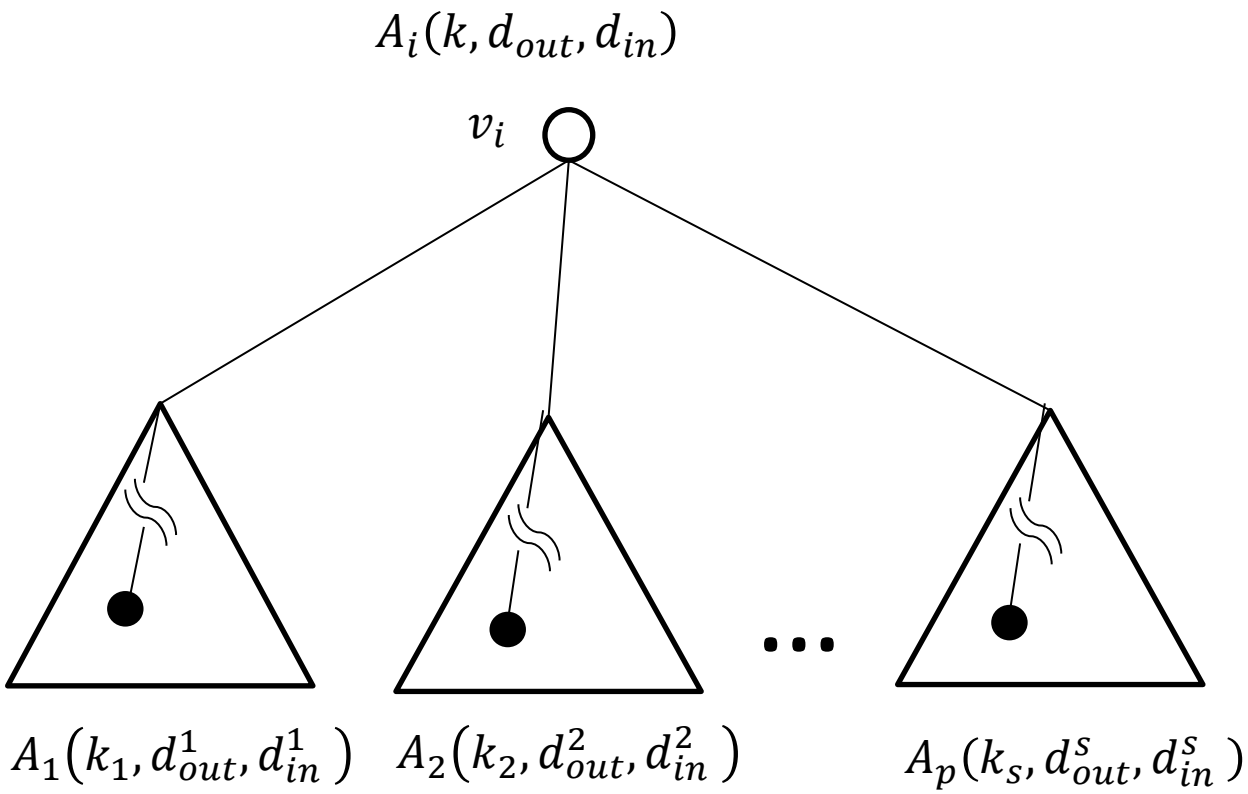
$$\boldsymbol{O\left(k^2 n^4\right)}$$

$A_i(k, d_{out}, d_{in})$

$$\max \begin{cases} \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^R \le n}} \{ A_L(j, d_{out}+1, d_{in}-1) + A_R\left(k-j, d_{out}+1, d_{in}^R\right)\} \\ \max\limits_{\substack{0 \le j \le k \\ d_{in}-1 \le d_{in}^L \le n}} \{A_L(j, d_{out}+1, d_{in}^L) + A_R(k-j, d_{out}+1, d_{in}-1)\} \end{cases}$$

s分木の場合

表のサイズは高々kn^2

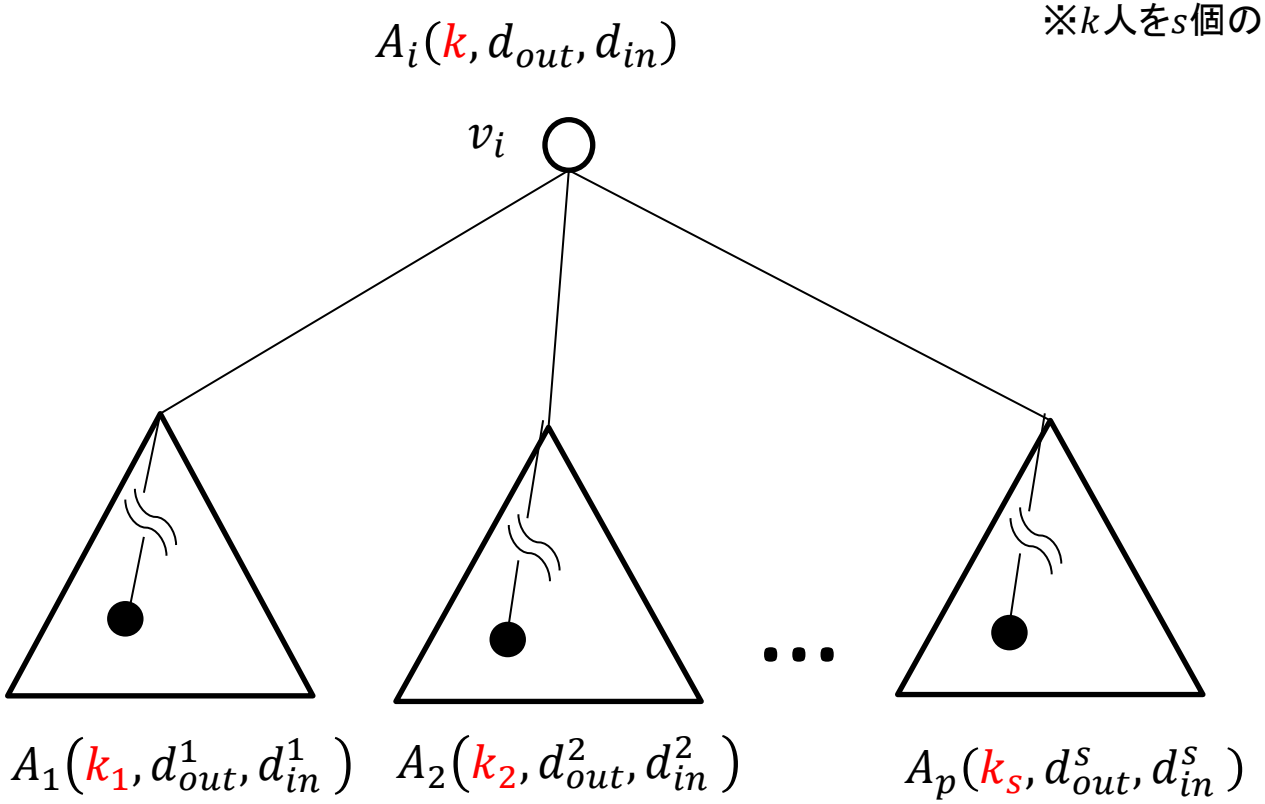


s分木の場合

表のサイズは高々kn^2

k_1～k_sの振り分け方が$\frac{(k+s-1)!}{k!(s-1)!}$通り

※k人をs個のグループに分ける場合の数に等しい



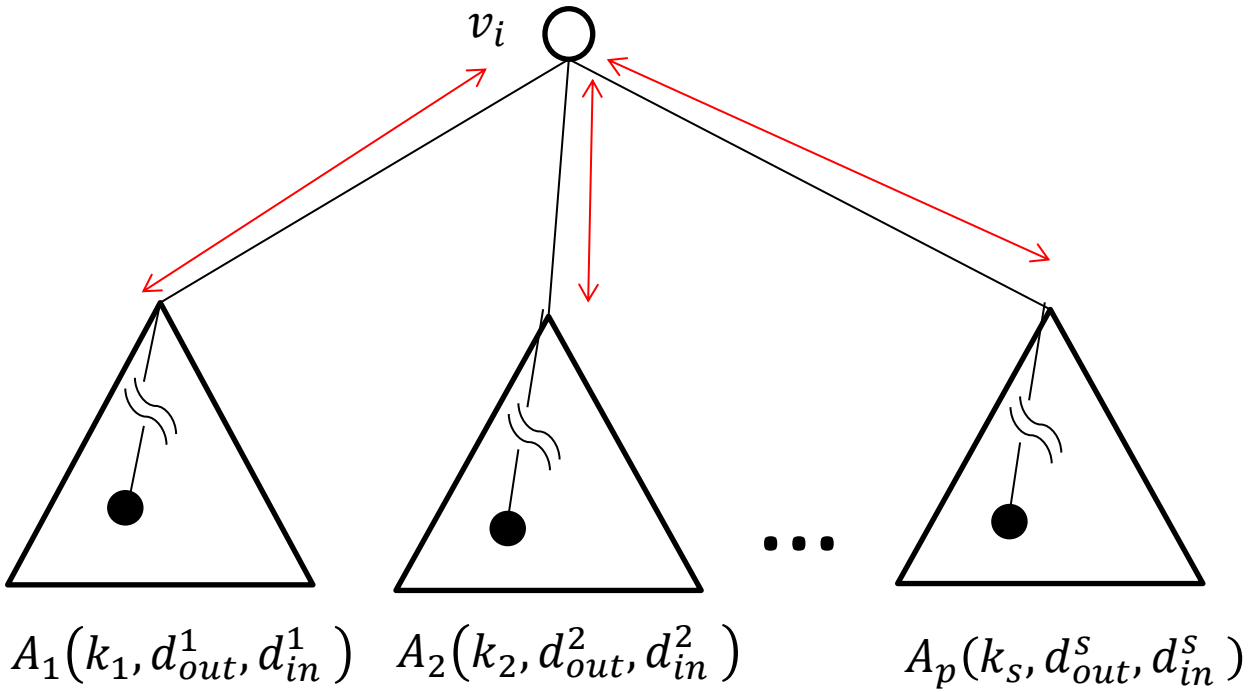
s分木の場合

表のサイズは高々kn^2

k_1～k_sの振り分け方が$\frac{(k+s-1)!}{k!(s-1)!}$通り

どこをd_{in}に定めるかでs通り

$A_i(k, d_{out}, d_{in})$



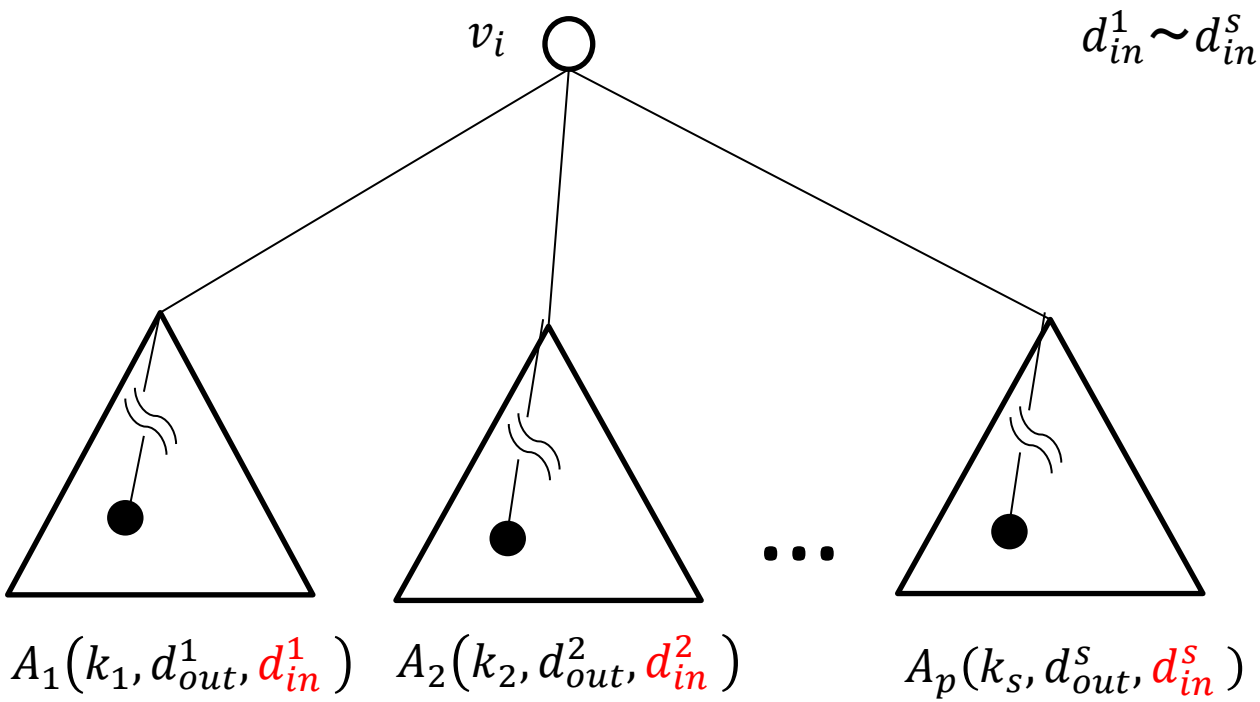
s分木の場合

表のサイズは高々kn^2

k_1～k_sの振り分け方が$\frac{(k+s-1)!}{k!(s-1)!}$通り

どこをd_{in}に定めるかでs通り

d_{in}^1～d_{in}^sのループが高々n^{s-1}回



s分木の場合

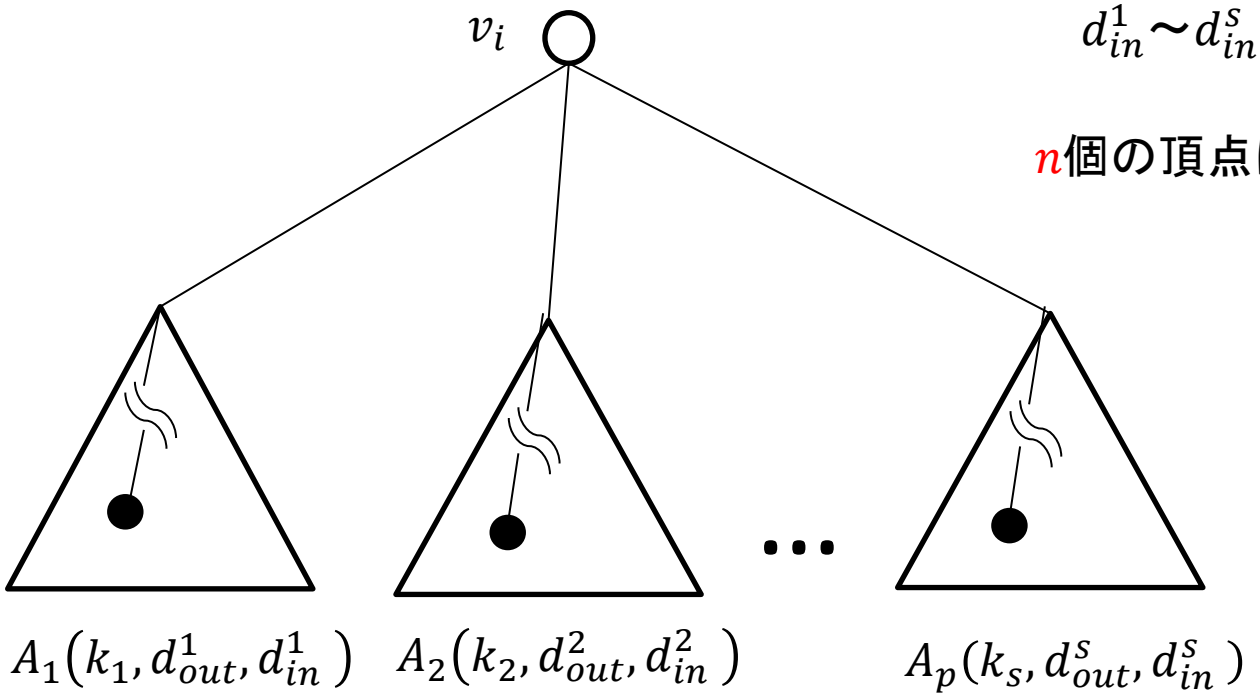
表のサイズは高々kn^2

k_1～k_sの振り分け方が$\frac{(k+s-1)!}{k!(s-1)!}$通り

どこをd_{in}に定めるかでs通り

d_{in}^1～d_{in}^sのループが高々n^{s-1}回

n個の頂点に対して$A_i(k, d_{out}, d_{in})$を求める



s分木の場合

計算時間

$$\boldsymbol{O\left(\frac{(k+s-1)!}{k!\,(s-1)!}\,ksn^{s+2}\right)}$$

表のサイズは高々kn^2

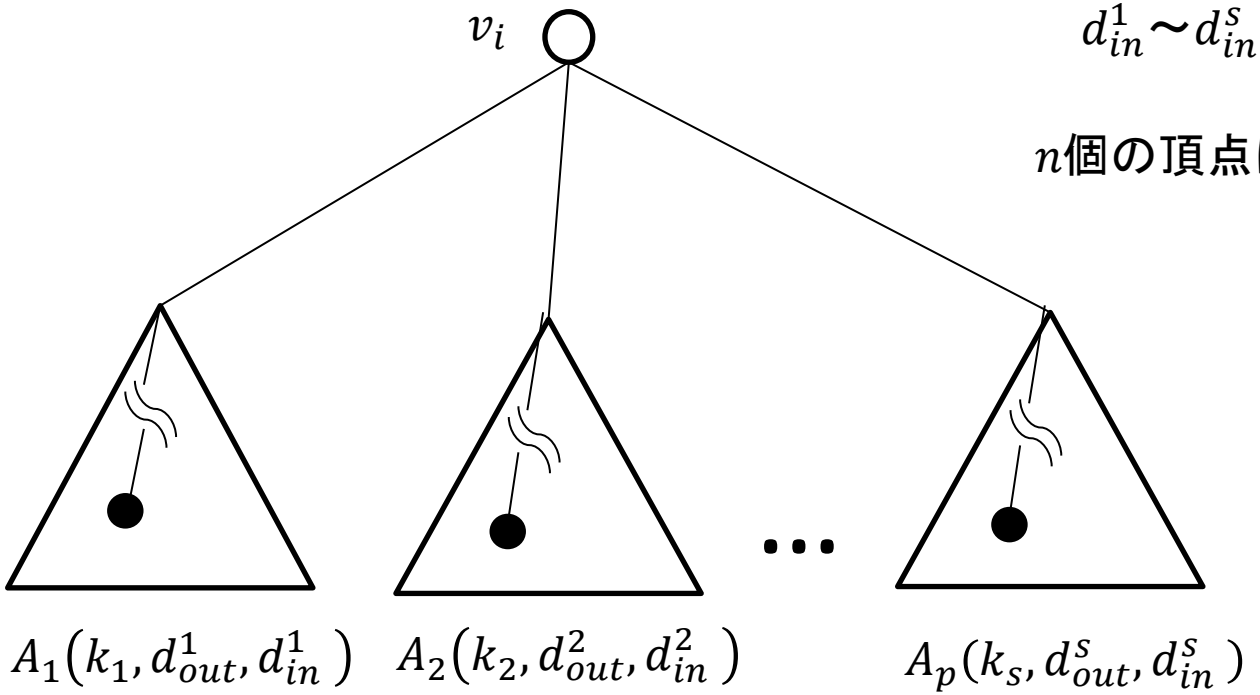
k_1～k_sの振り分け方が$\frac{(k+s-1)!}{k!(s-1)!}$通り

どこをd_{in}に定めるかでs通り

d_{in}^1～d_{in}^sのループが高々n^{s-1}回

n個の頂点に対して$A_i(k, d_{out}, d_{in})$を求める

$A_i(k, d_{out}, d_{in})$



最後に…

まとめ

木の(1,k)-ボロノイゲームにおいて、先手のk個の頂点が与えられたとき、(後手)−(先手)の値の最大値を求めるアルゴリズムを示した

二分木の場合

計算時間: $\mathcal{O}(k^2 n^4)$

s分木の場合

計算時間: $\mathcal{O}\left(\frac{(k+s-1)!}{k!\,(s-1!)} k s n^{s+2}\right)$

今後の課題

・アルゴリズムの改良

・木の(1,k)-ボロノイゲームにおいて、後手に勝つ手が存在するかどうかを考える