消防士問題に対する モジュラ幅FPTアルゴリズム

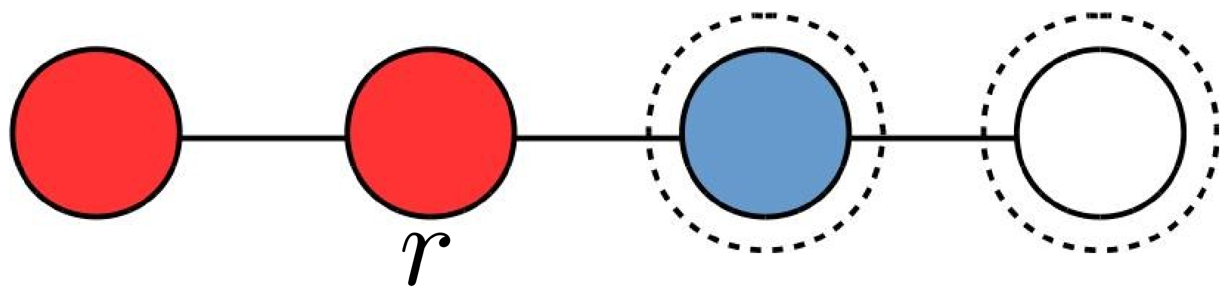
2018年3月6日
組合せゲーム･パズル研究集会
大阪府立大学

吉村純弥[†], 清見礼[‡], 大舘陽太[†]
† 熊本大学
‡ 横浜市立大学

Firefighting problemのルール

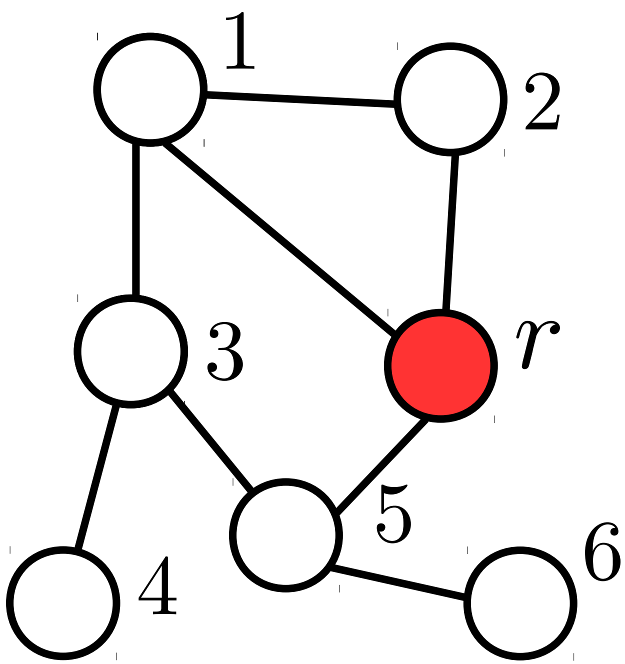
入力：グラフ $G=(V,E)$ と頂点 $r \in V$

- 時刻 0 で r に火が点く.
- 以降の各時刻で, 火と隣接する全頂点に火が点く.
- 各時刻と次の時刻の間で,
  火の点いていない1つの頂点を守ることができる.
- 守られた頂点にそれ以降火は点かない.

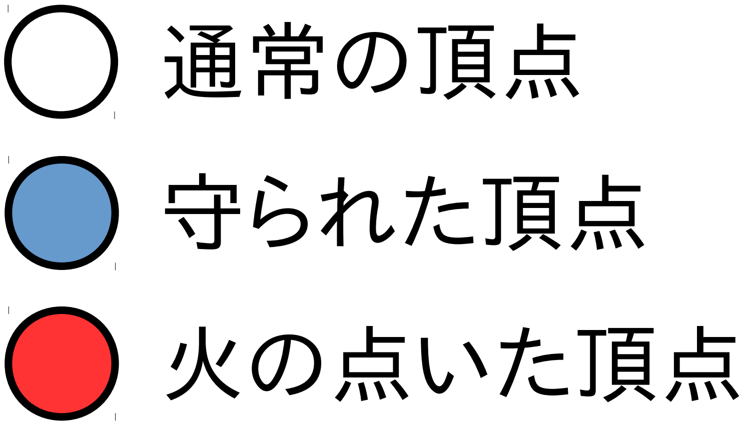


目的：燃えなかった頂点数の最大化

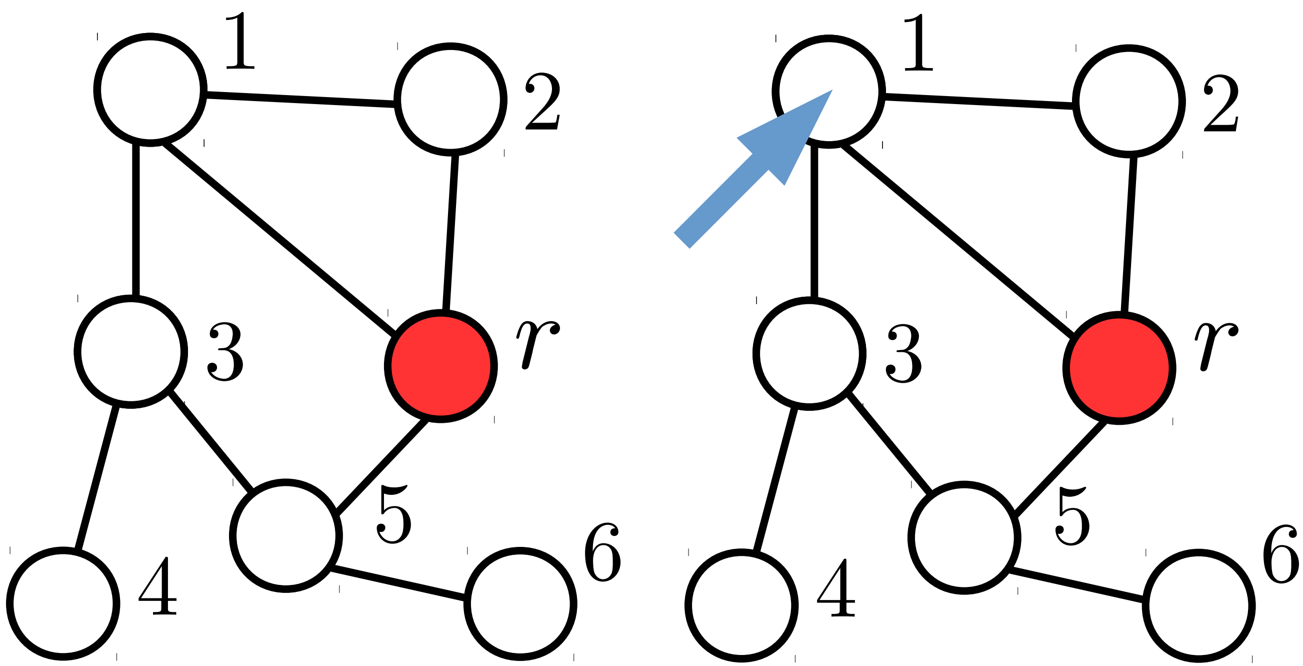
Firefighting problem の戦略例



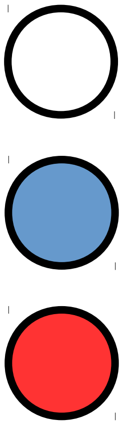
$t = 0$



Firefighting problem の戦略例

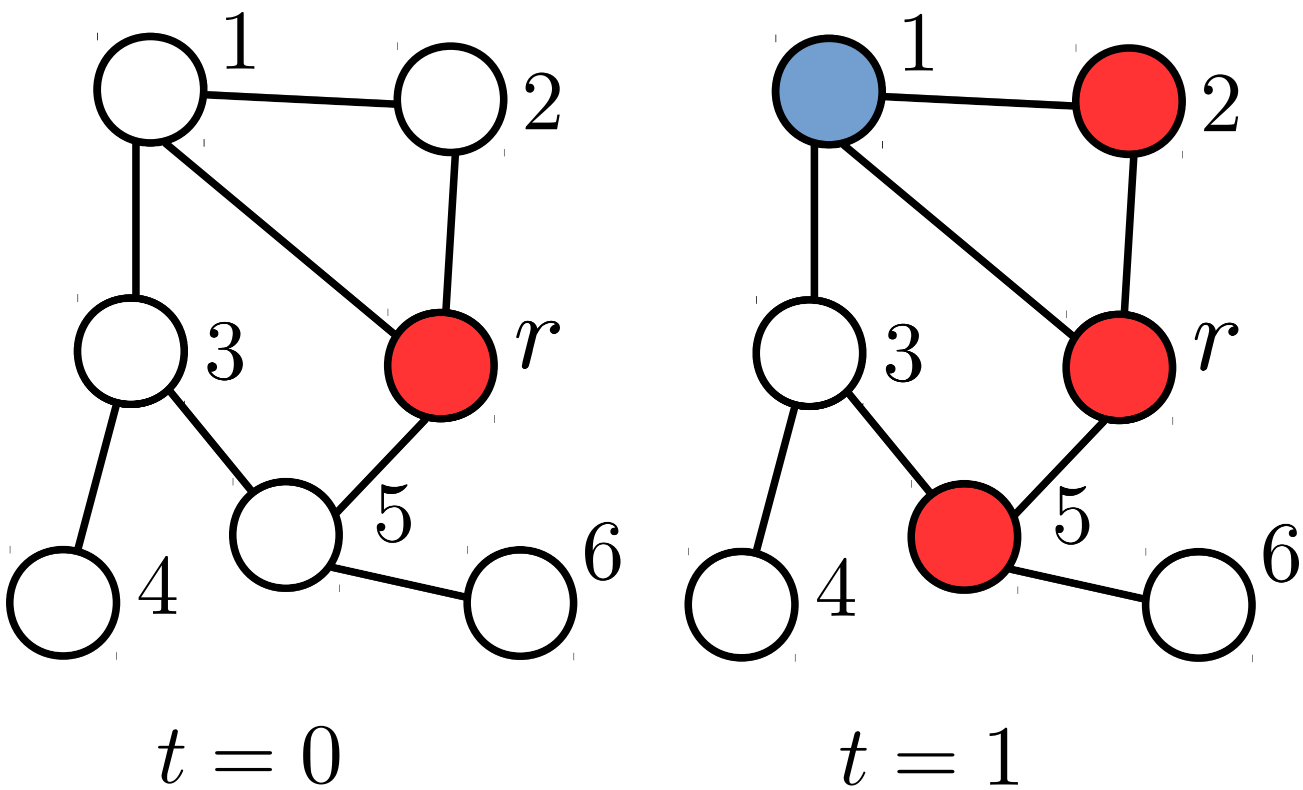


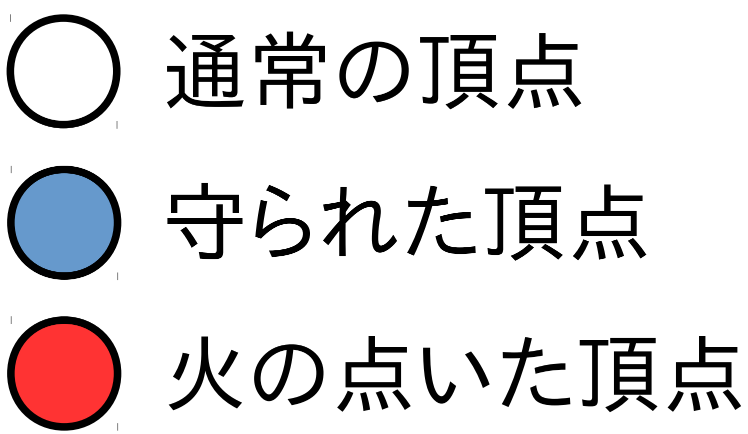
$t = 0$



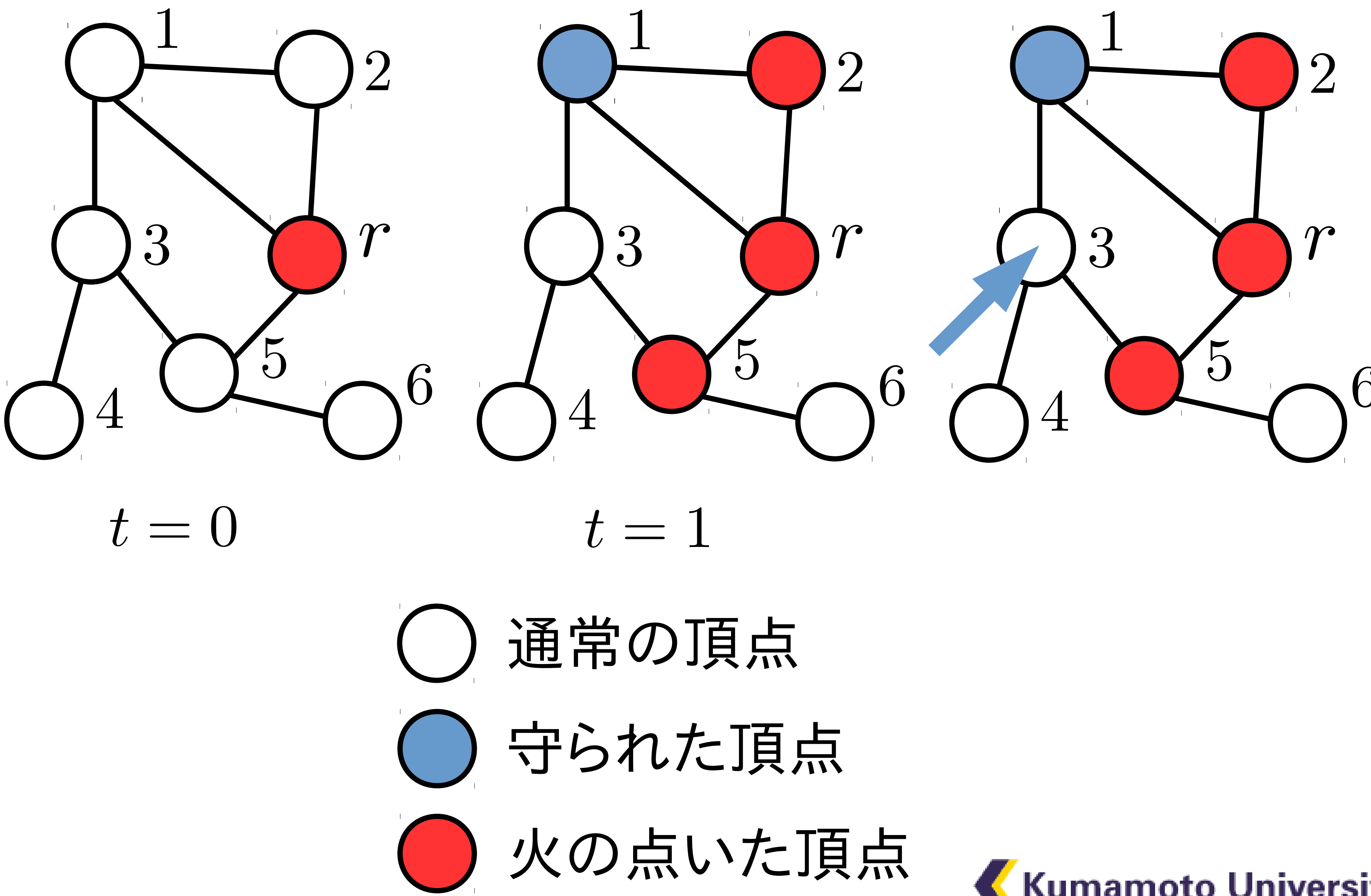
Kumamoto University

Firefighting problem の戦略例



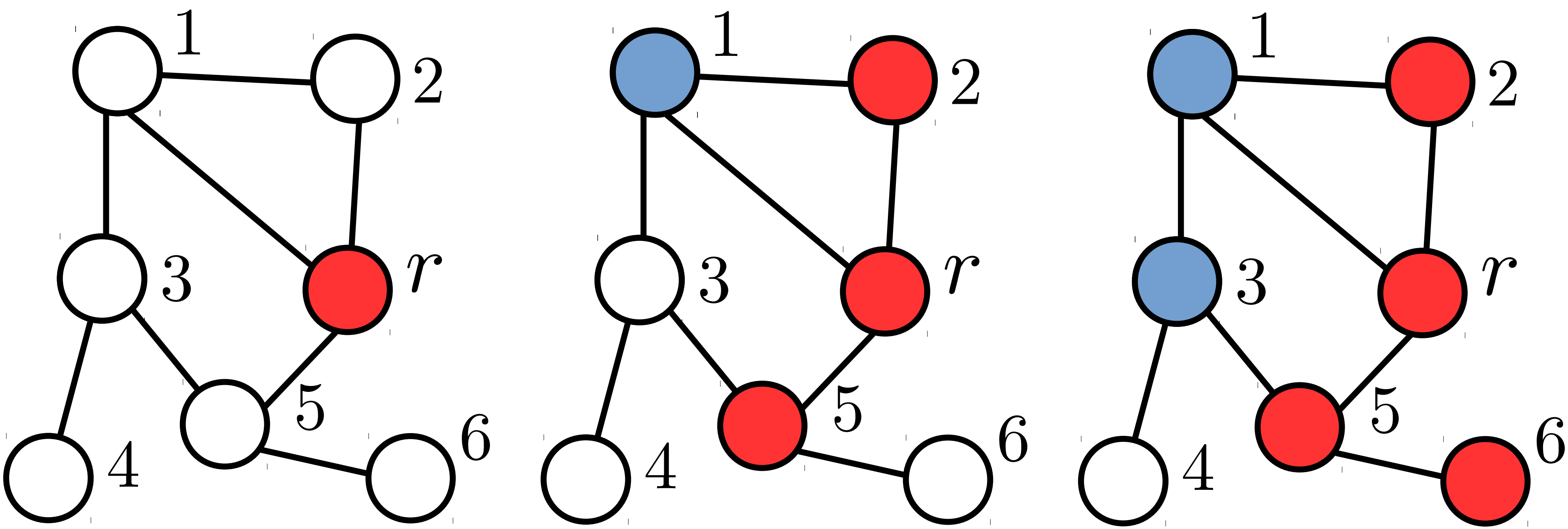


Firefighting problem の戦略例



Kumamoto University

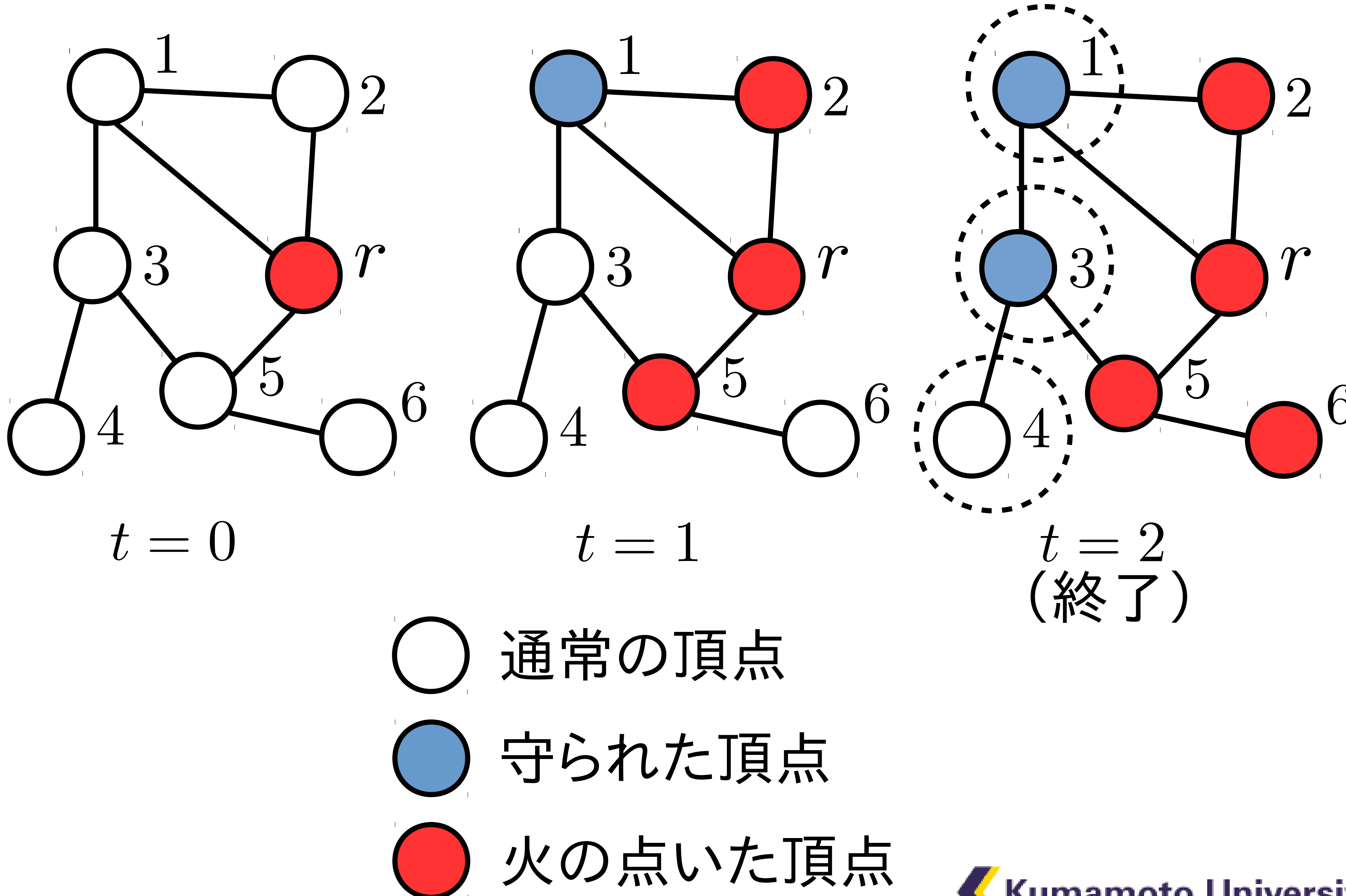
Firefighting problem の戦略例



$t = 0$　　$t = 1$　　$t = 2$

- 通常の頂点
- 守られた頂点
- 火の点いた頂点

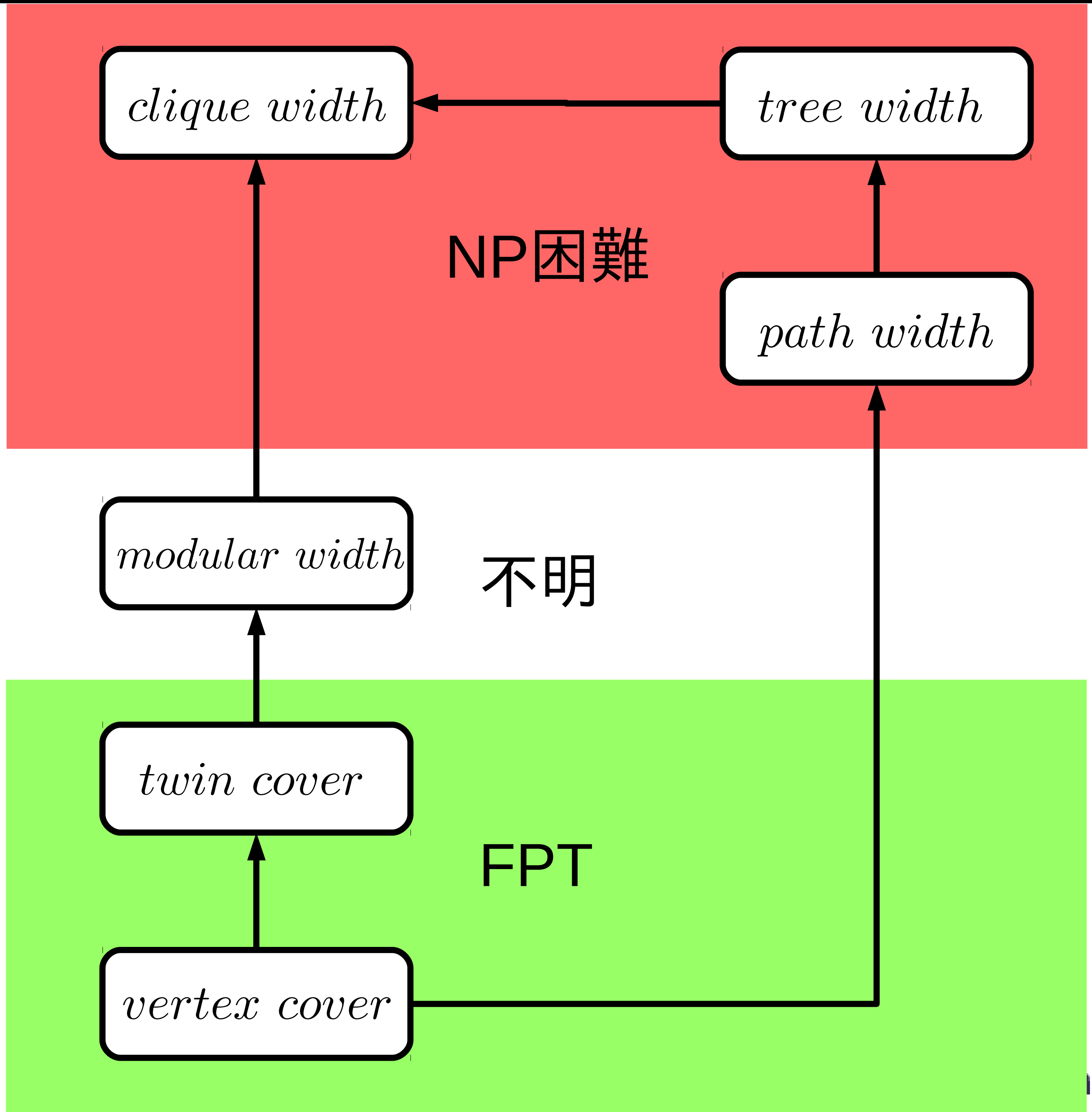
Firefighting problem の戦略例



本研究の動機

- 最適解を計算するアルゴリズムを作りたい
- 計算量理論的に難しい（NP困難）
- グラフがよい構造を持つ場合を考える

Firefighting problem の現状



本研究の結果

- modular widthがkであるグラフ$G=(V,E)$を入力としたFirefighting problemに対するFPTアルゴリズムの設計
  - ・ $O(2^k \cdot k \cdot (|E|+|V|))$ 時間
- 先行研究であるtwin coverを用いたFirefighting problemに対するFPTアルゴリズム[Robert Ganian, 2015]の修正
  - ・場合分けの見落としがあった

Kumamoto University

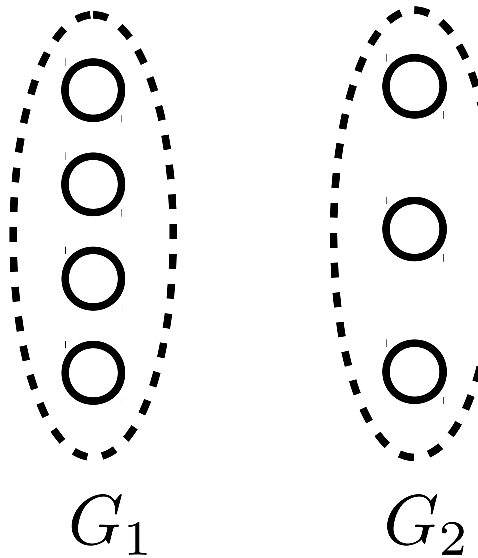
本研究の結果

- modular widthが k であるグラフ $G=(V,E)$ を入力としたFirefighting problemに対するFPTアルゴリズムの設計
  - $O(2^k \cdot k \cdot (|E|+|V|))$ 時間
- 先行研究であるtwin coverを用いたFirefighting problemに対するFPTアルゴリズム[Robert Ganian, 2015]の修正
  - ・場合分けの見落としがあった

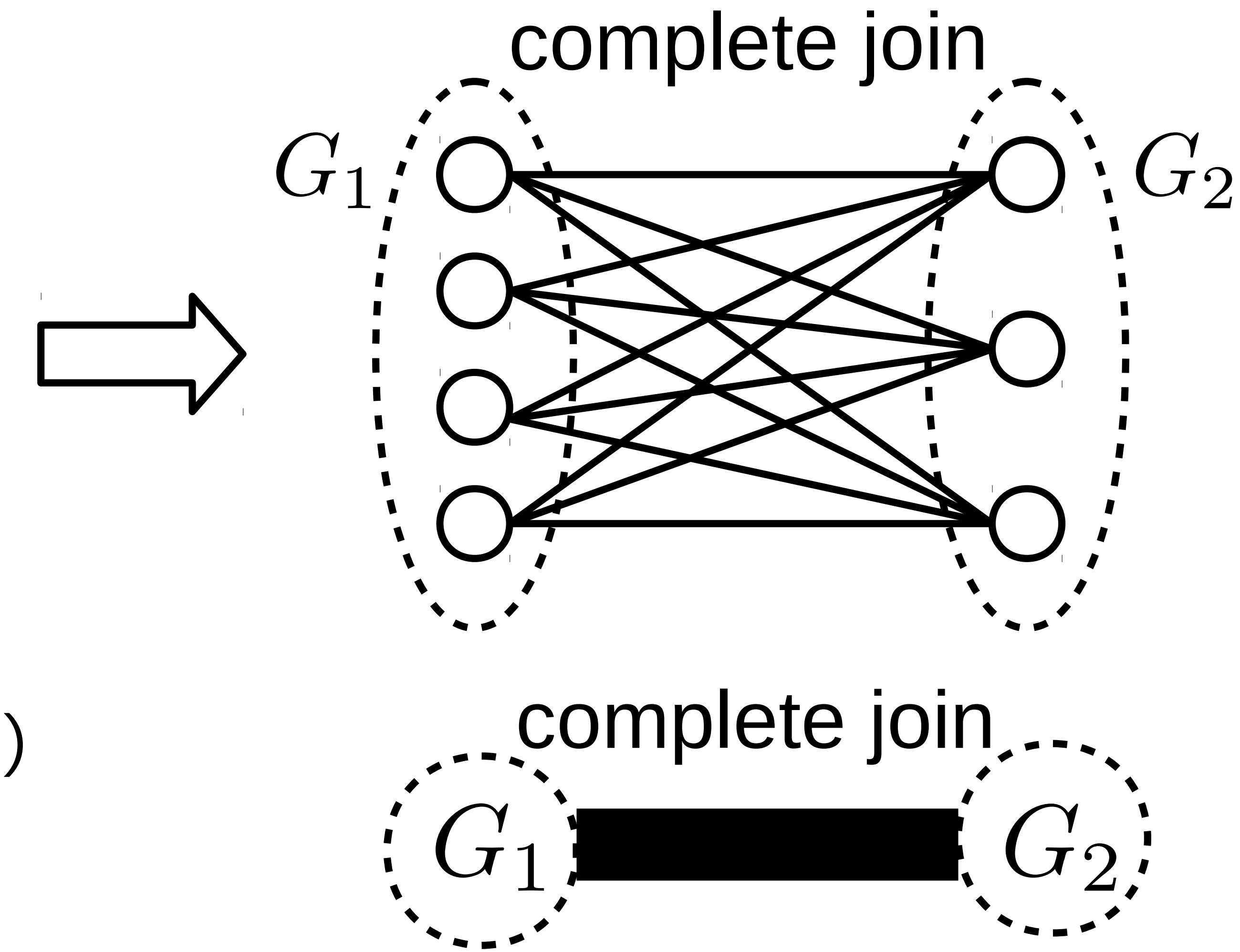
Kumamoto University

定義　complete join

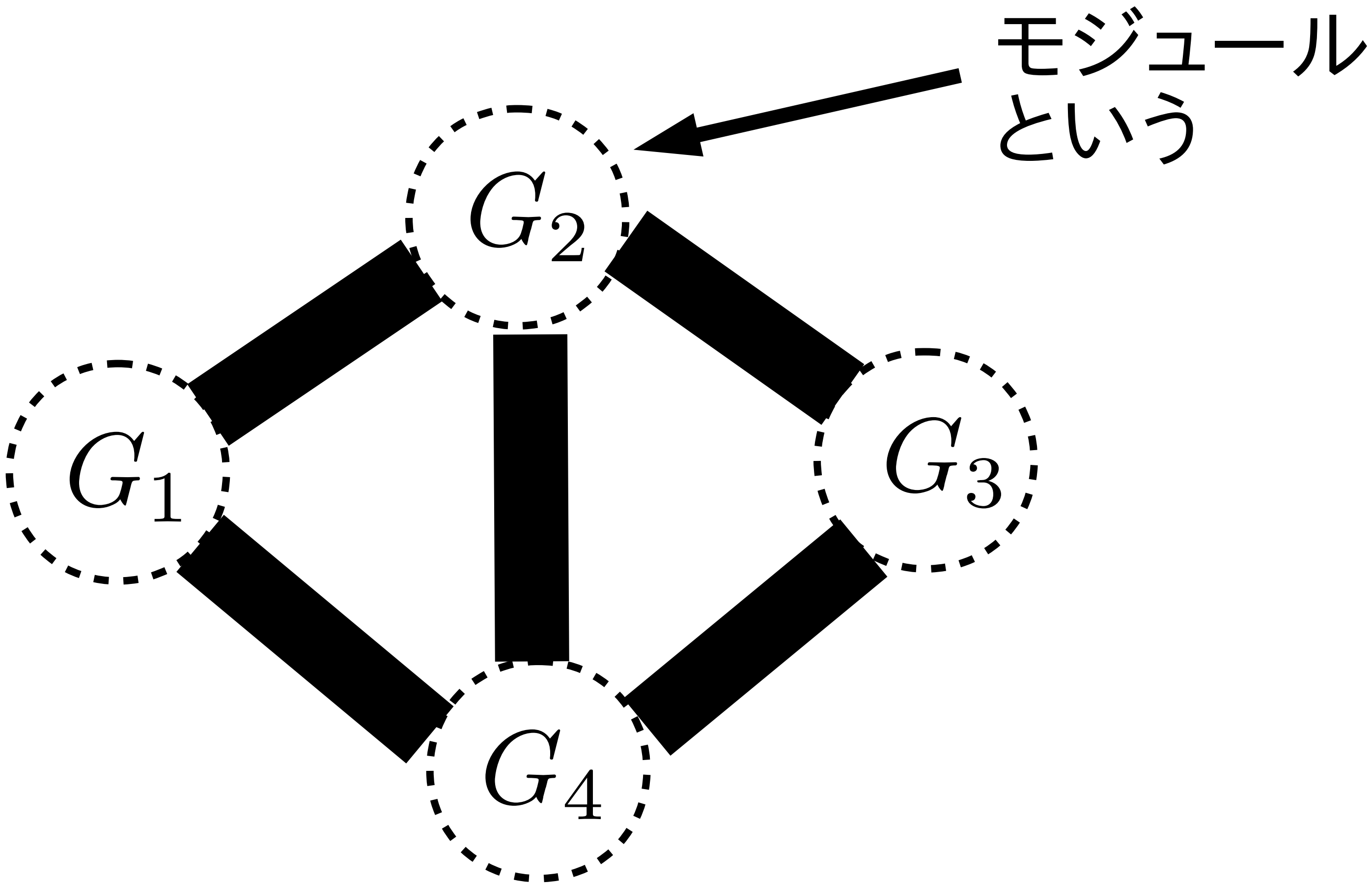
グラフ G_1 と G_2 のcomplete join



（各 G_i 内の辺は省略）



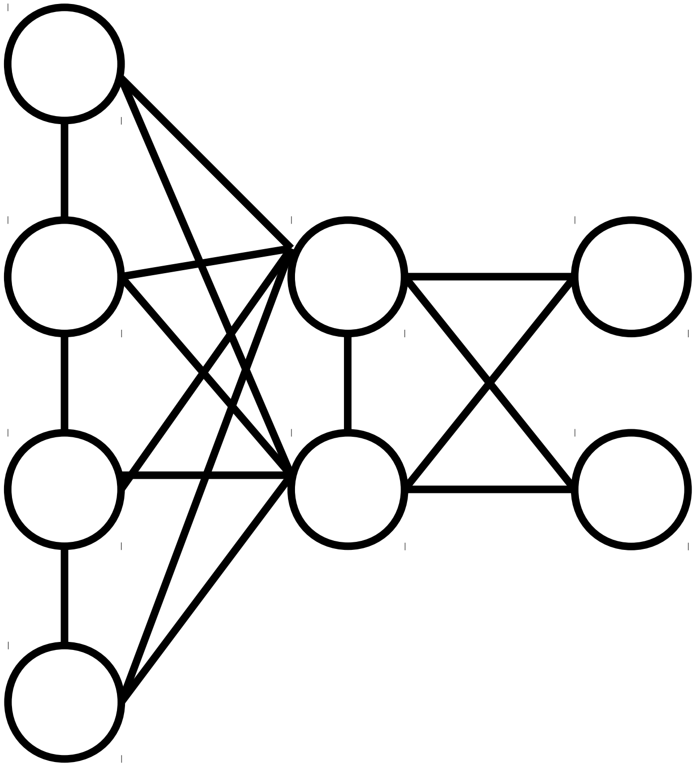
定義　モジュールへの分解



幅4の分解

モジュールへの分解の例

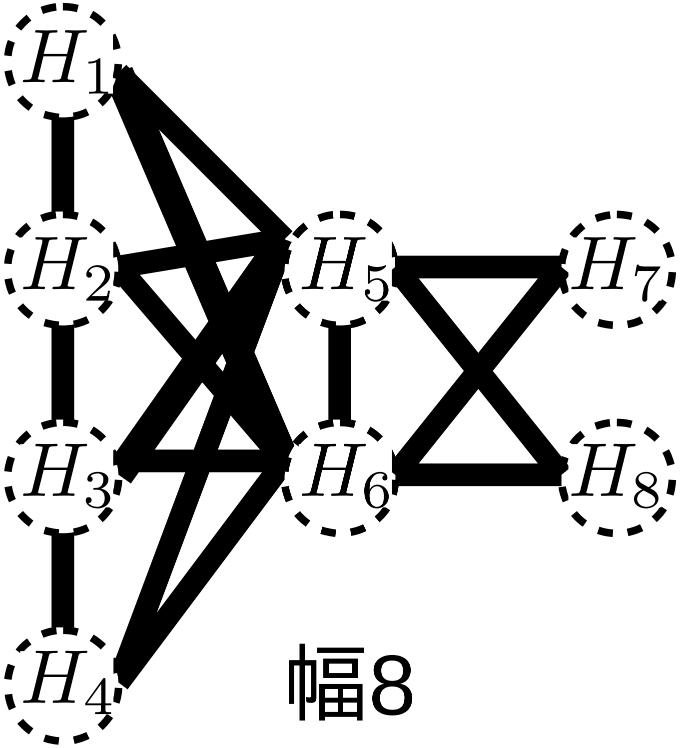
1つのグラフに対して様々なモジュールへの分解が存在.



例1

幅3

例2

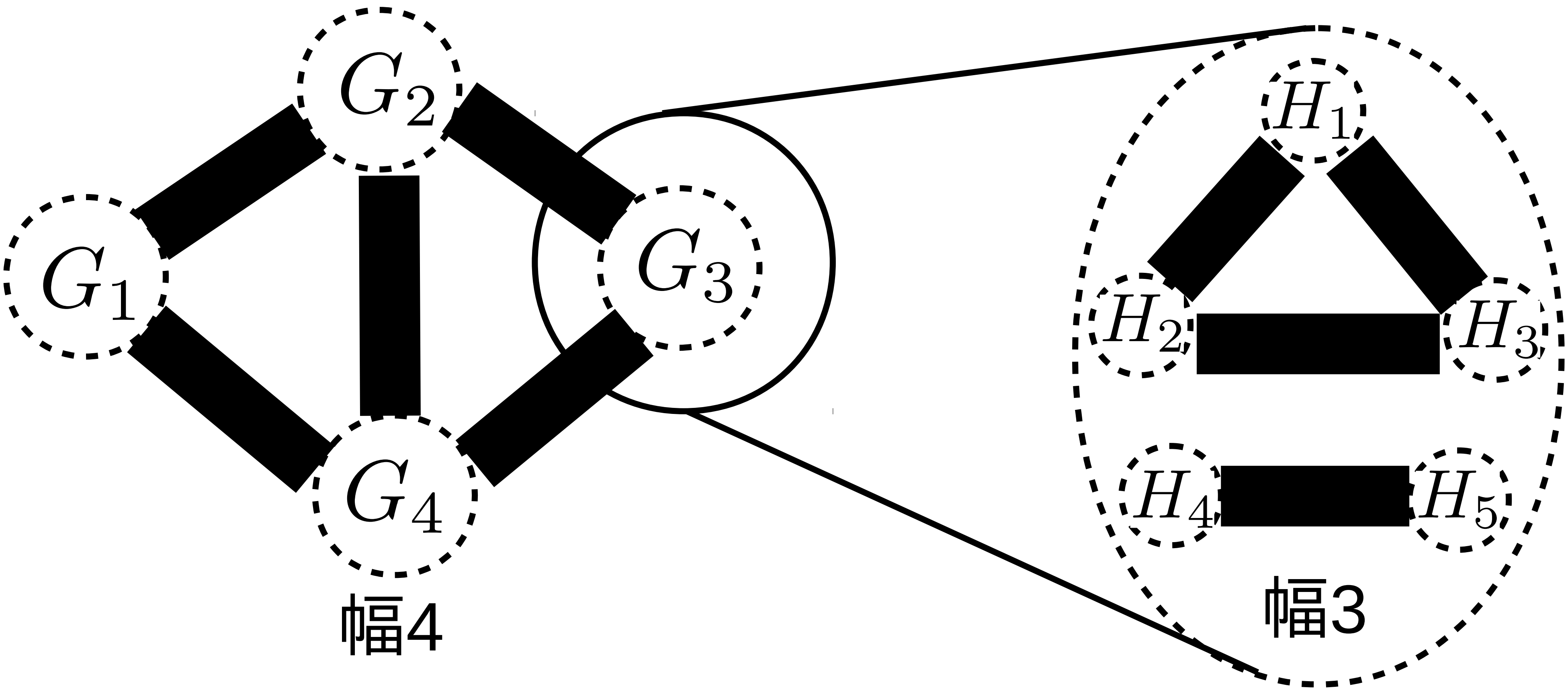


幅8

Kumamoto University

定義　modular width

幅 k 以下のモジュールへの分解を, 各モジュールに対して 1 頂点になるまで再帰的に行えるような最小の k をmodular widthという.



modular widthの計算時間

命題1 [Tedder, Corneil, Habib, Paul. 2008.]

$G=(V,E)$ について考える. Gのmodular widthを計算し, さらに幅 $mw(G)$以下のモジュールへの分解を, 各モジュールに対して1頂点になるまで再帰的に行うことは, $O(|E|+|V|)$ 時間で実現できる.

※　グラフGのmodular widthを $mw(G)$ で表記.

問題に対する大まかなアルゴリズム

命題2 [Fomin, Heggernes, van Leeuwen. 2016.]

$G=(V,E)$ を入力したFirefighting problemを考える. ある戦略が有効であるかどうかを確認し, かつ有効であればその戦略の解を求めることを $O(|E|+|V|)$ 時間で行うことができる.

全ての戦略に対して命題2のアルゴリズムを適用

Kumamoto University

戦略の総数

- ある時刻の間で, 最大 $|V|$ 個の頂点の守り方が存在.
- 火は最大で $|V|-1$ 回広がる.

戦略の総数　：　守る頂点の候補数$^{\text{火が広がる回数}}$（通り）

$$\leq |V|^{|V|-1}$$

火の広がる最大回数

定理1(証明略)

誘導パスの最大の長さが x であるグラフについてFirefighting problemを考えたとき, 火は最大で x 回広がる.

定理2（証明略）

modular widthが $k \geq 3$ であるグラフの誘導パスの最大の長さは $k-1$ 以下である.

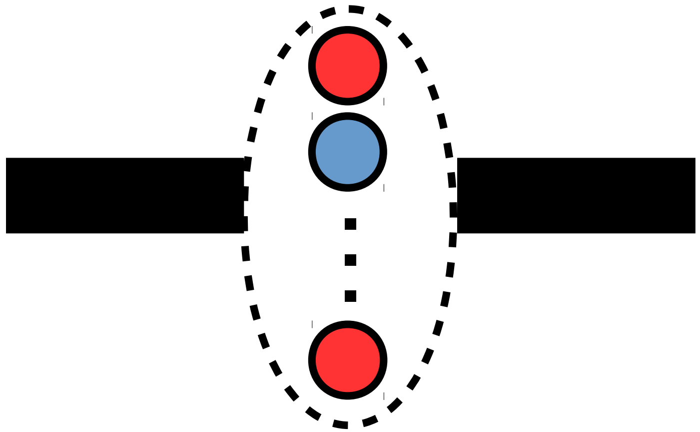
- 火は最大で $k-1$ 回広がる.

$$|V|^{k-1} \quad \text{通りの戦略}$$

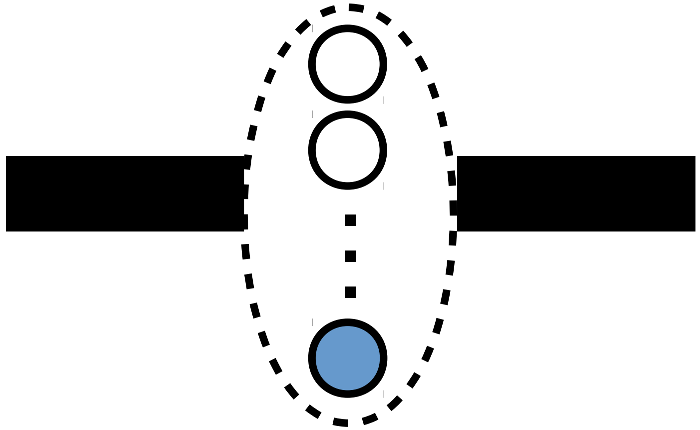
Kumamoto University

rを含まないモジュールの観察

- 守られていない全ての頂点に火が点く.



- または, 全ての頂点に火は点かない.



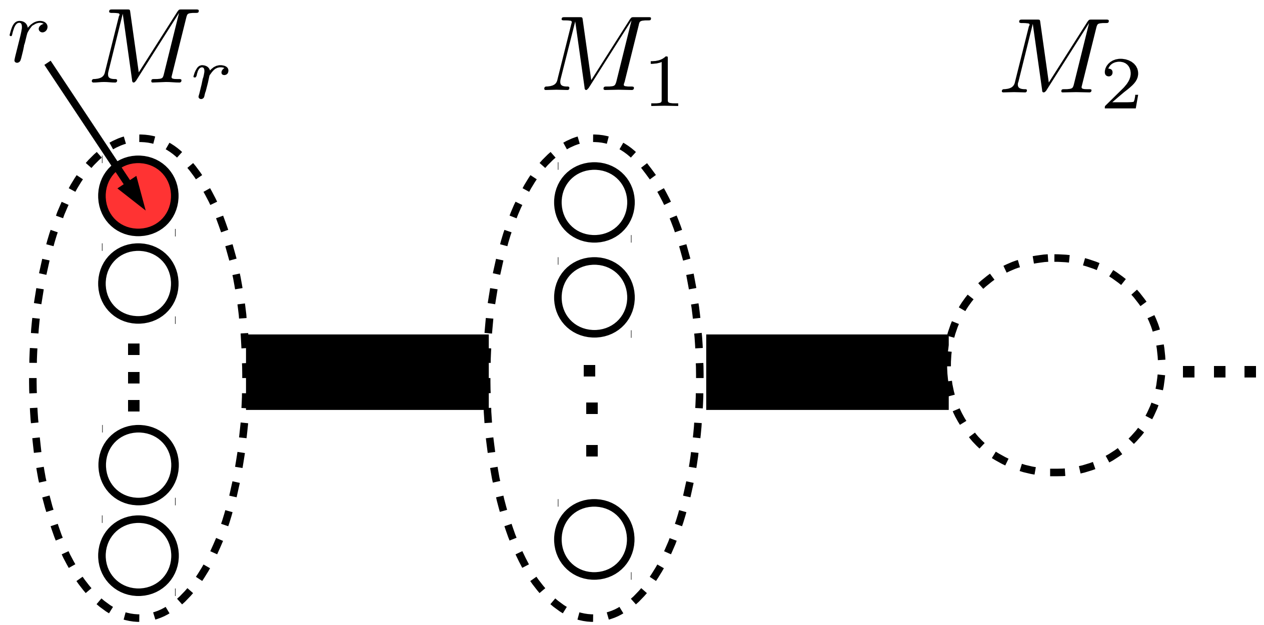
r を含まないあるモジュール内であれば
どの頂点を守っても同じ.

Kumamoto University

rを含むモジュールの観察

M_r を r を含むモジュールとする

- M_r 以外のモジュールの頂点に火が点くとき



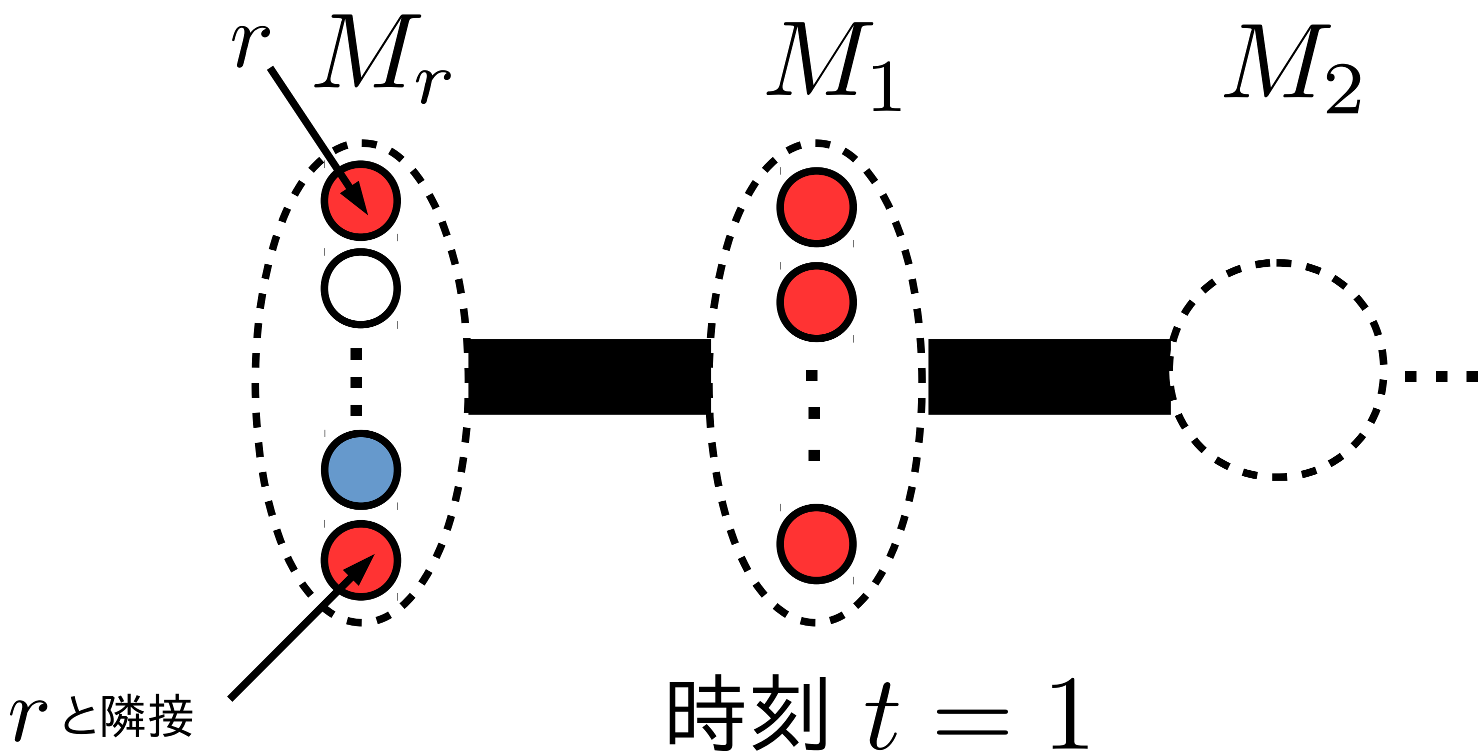
時刻 $t = 0$

Kumamoto University

rを含むモジュールの観察

M_r を r を含むモジュールとする

- M_r 以外のモジュールの頂点に火が点くとき



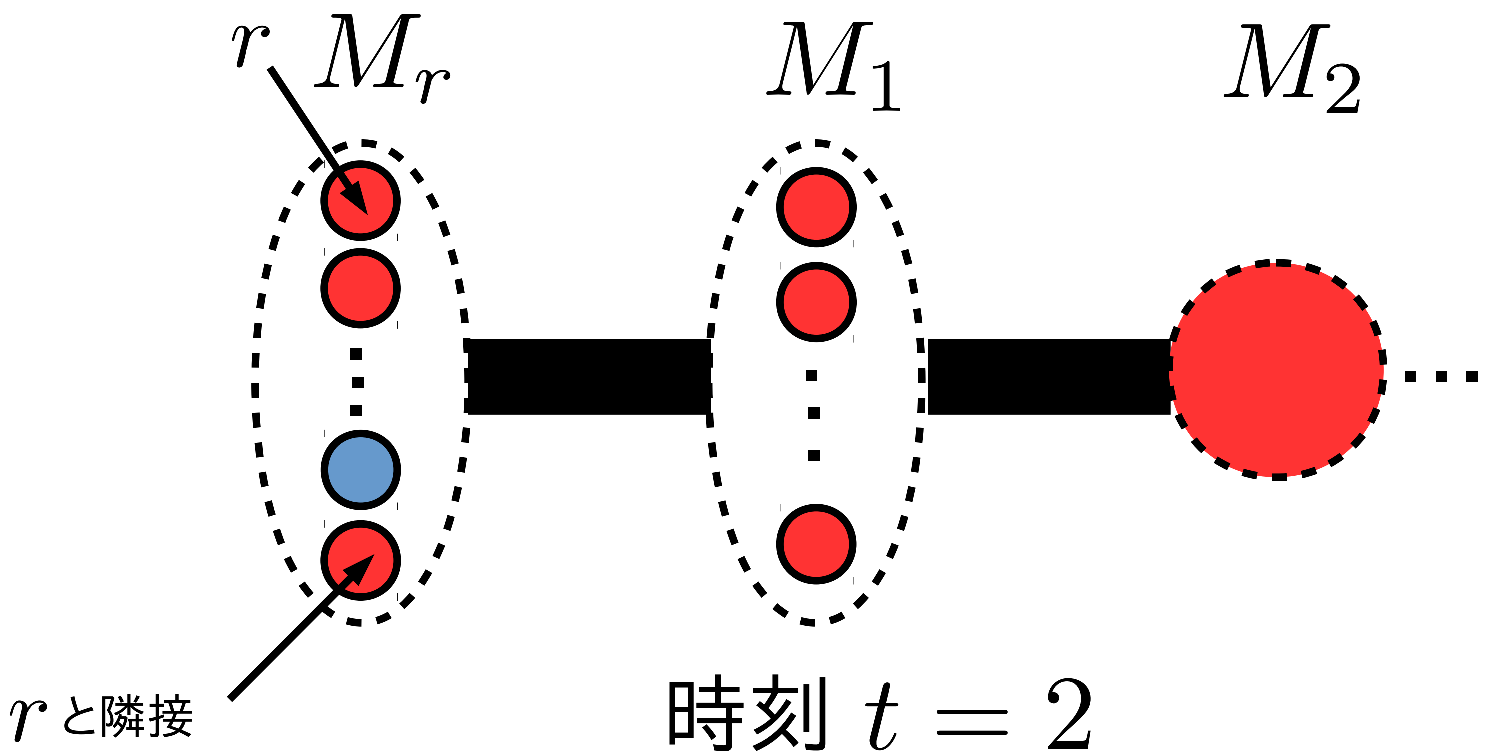
時刻 $t = 1$

Kumamoto University

rを含むモジュールの観察

M_r を r を含むモジュールとする

- M_r 以外のモジュールの頂点に火が点くとき

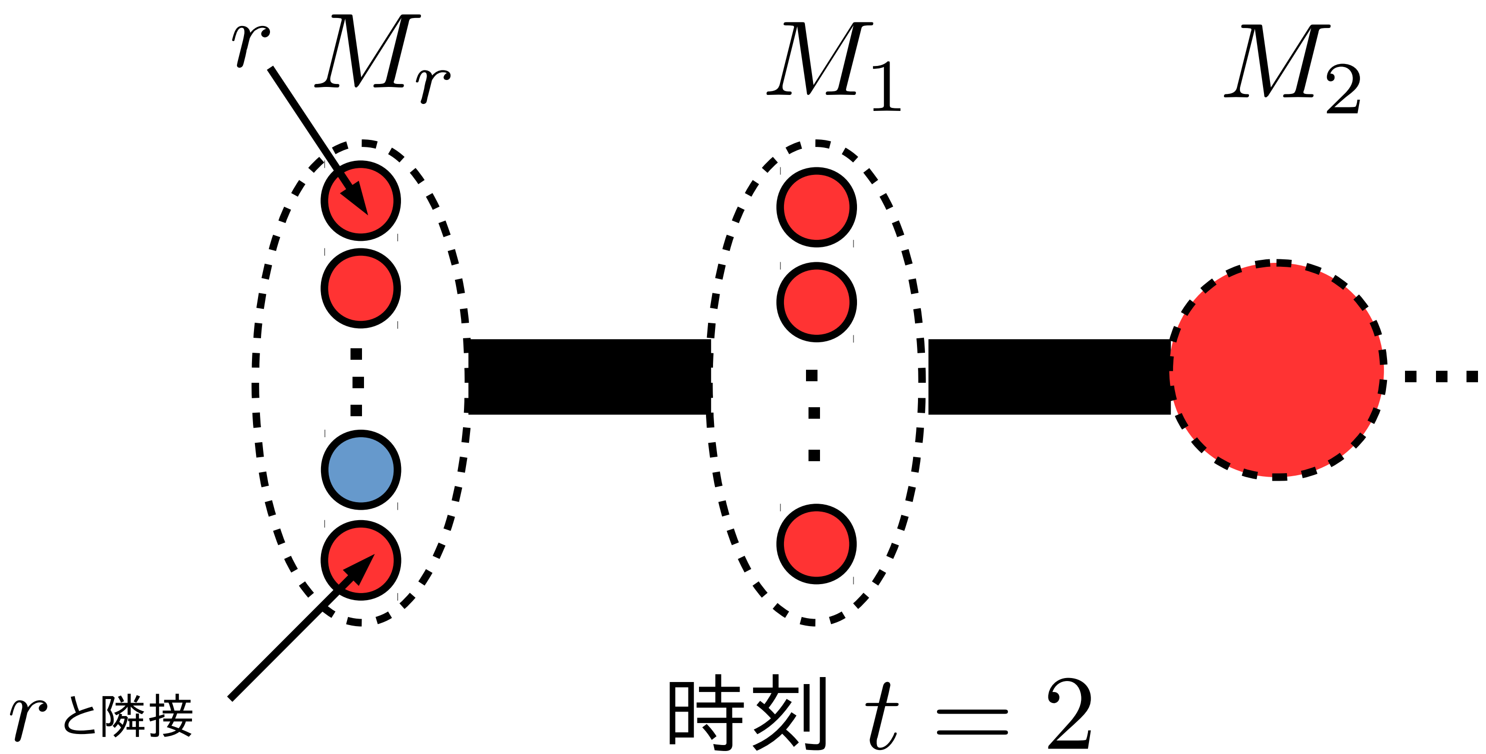


時刻 $t = 2$

rを含むモジュールの観察

M_r を r を含むモジュールとする

- M_r 以外のモジュールの頂点に火が点くとき

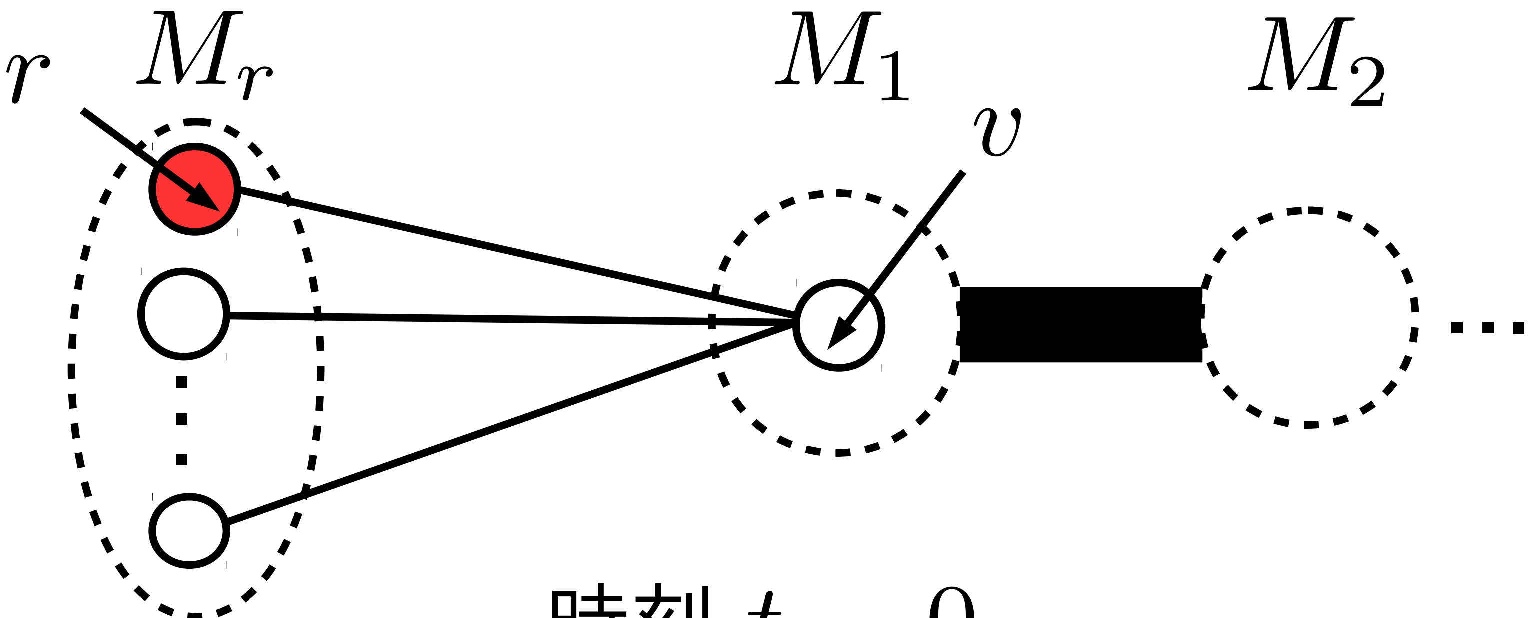


時刻 $t = 2$

M_r の守られていない全ての頂点に火が点く

rを含むモジュールの観察

- M_r以外のモジュールの頂点に火が点かないとき
  - ➔ rと隣接し, かつM_rに属さない頂点vが1つだけ存在
  - ➔ $t=0$と$t=1$の間でvを守る

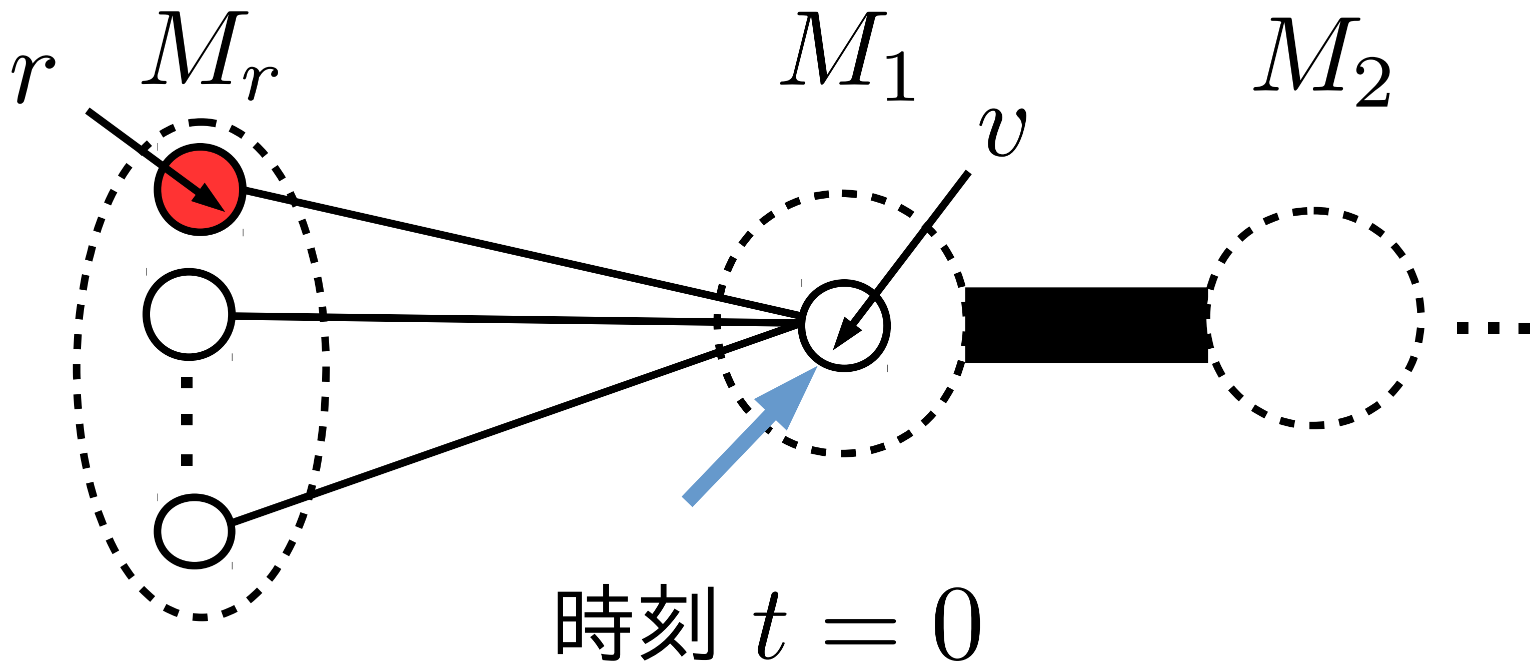


時刻 $t=0$

Kumamoto University

rを含むモジュールの観察

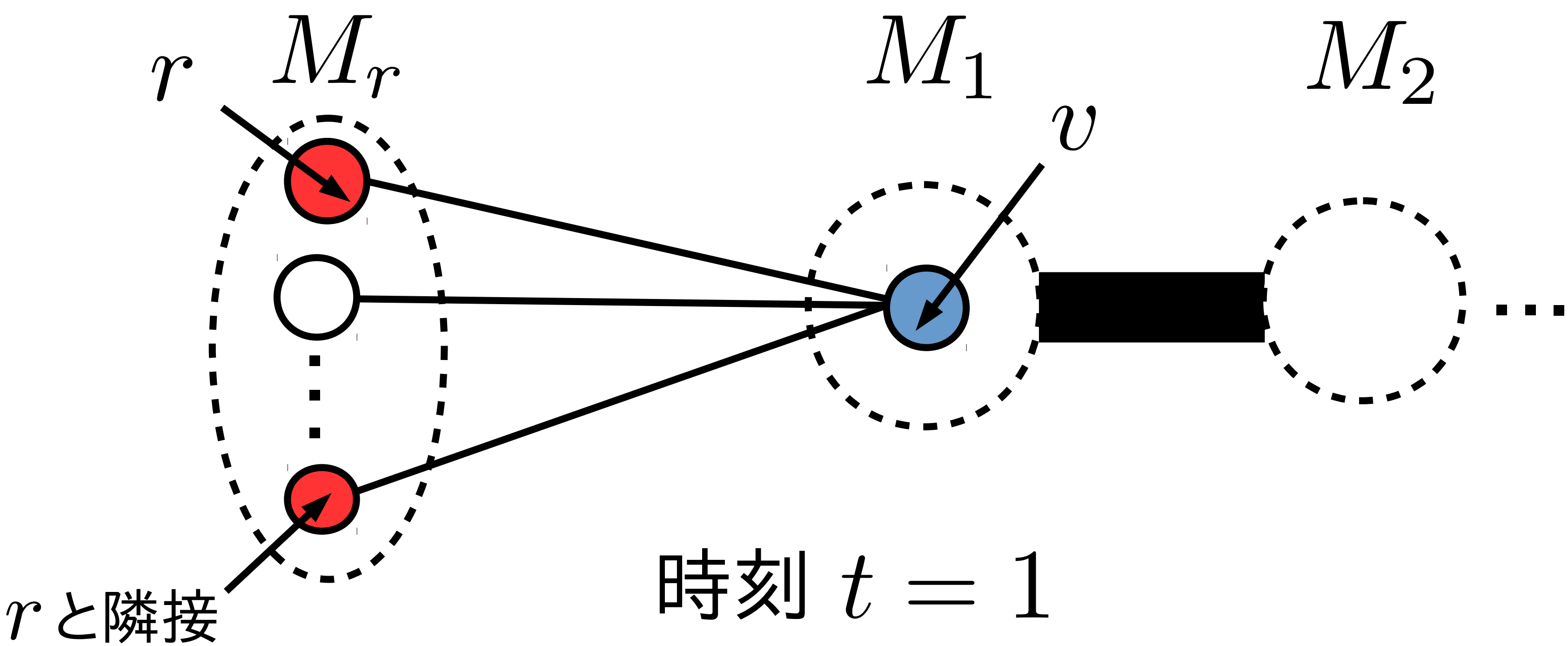
- M_r 以外のモジュールの頂点に火が点かないとき
  - ➔ r と隣接し，かつ M_r に属さない頂点 v が1つだけ存在
  - ➔ $t=0$ と $t=1$ の間で v を守る



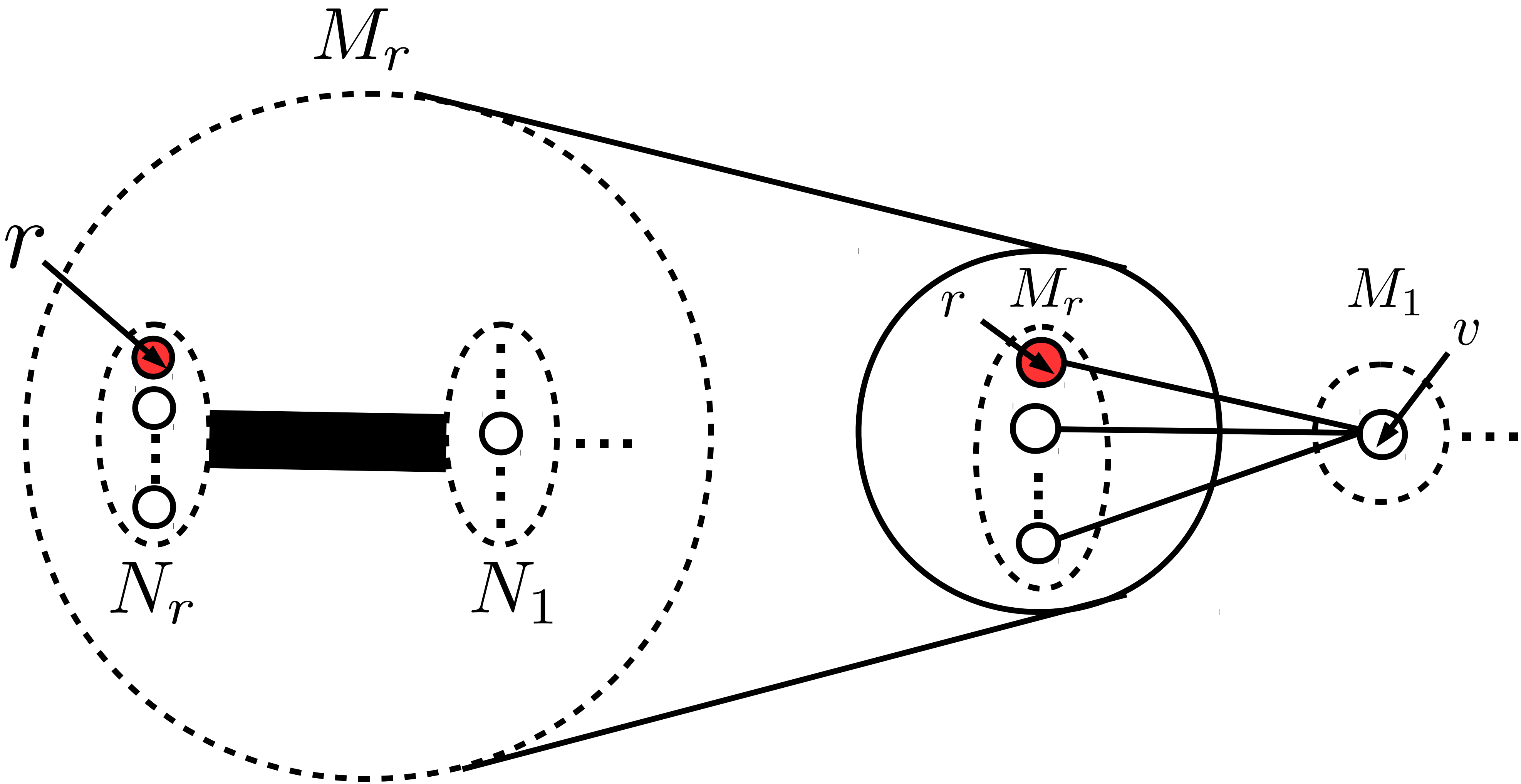
Kumamoto University

rを含むモジュールの観察

- M_r 以外のモジュールの頂点に火が点かないとき
  - ➔ r と隣接し, かつ M_r に属さない頂点 v が 1つだけ存在
  - ➔ $t=0$ と $t=1$ の間で v を守る

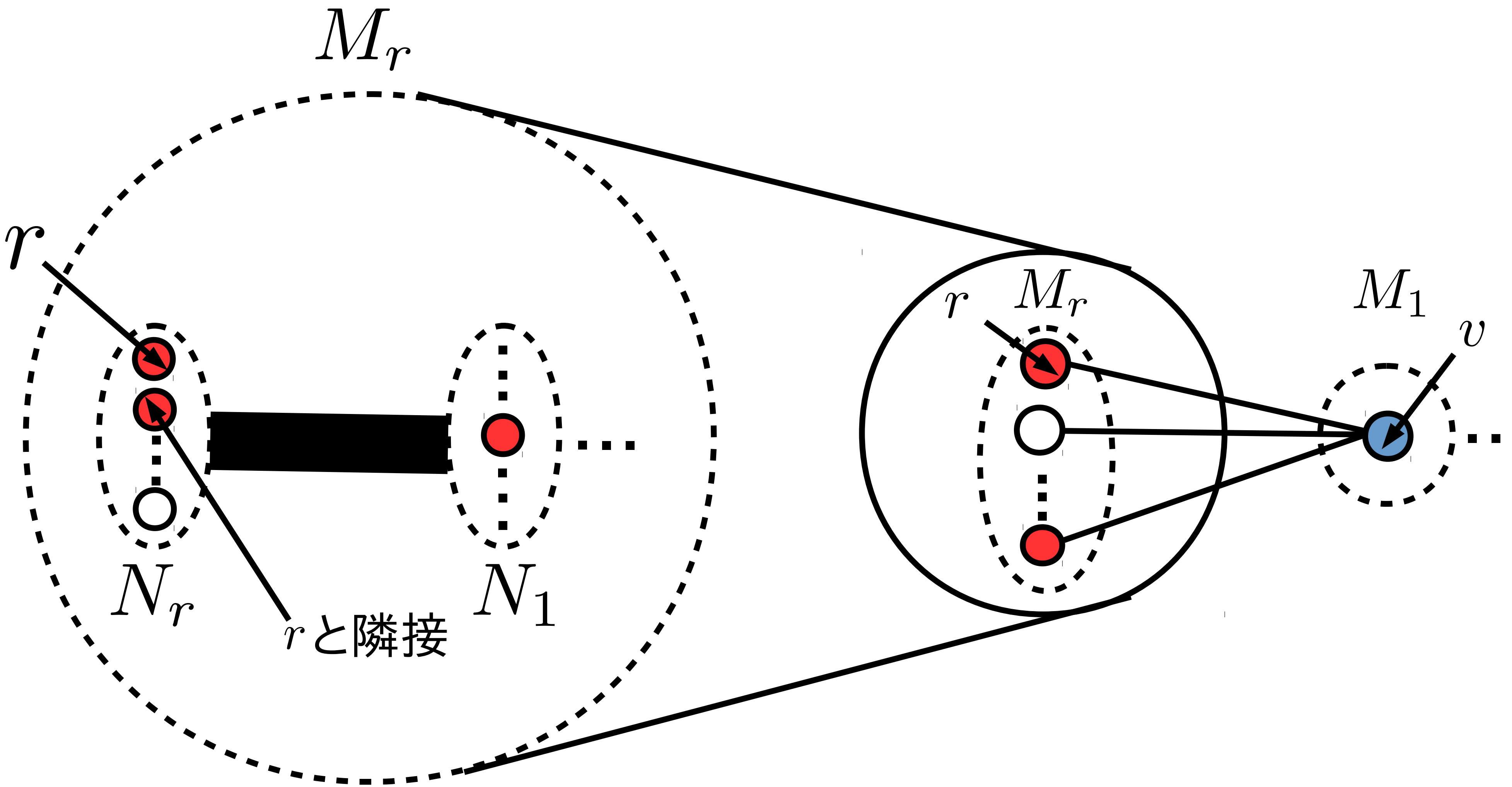


rを含むモジュールの観察



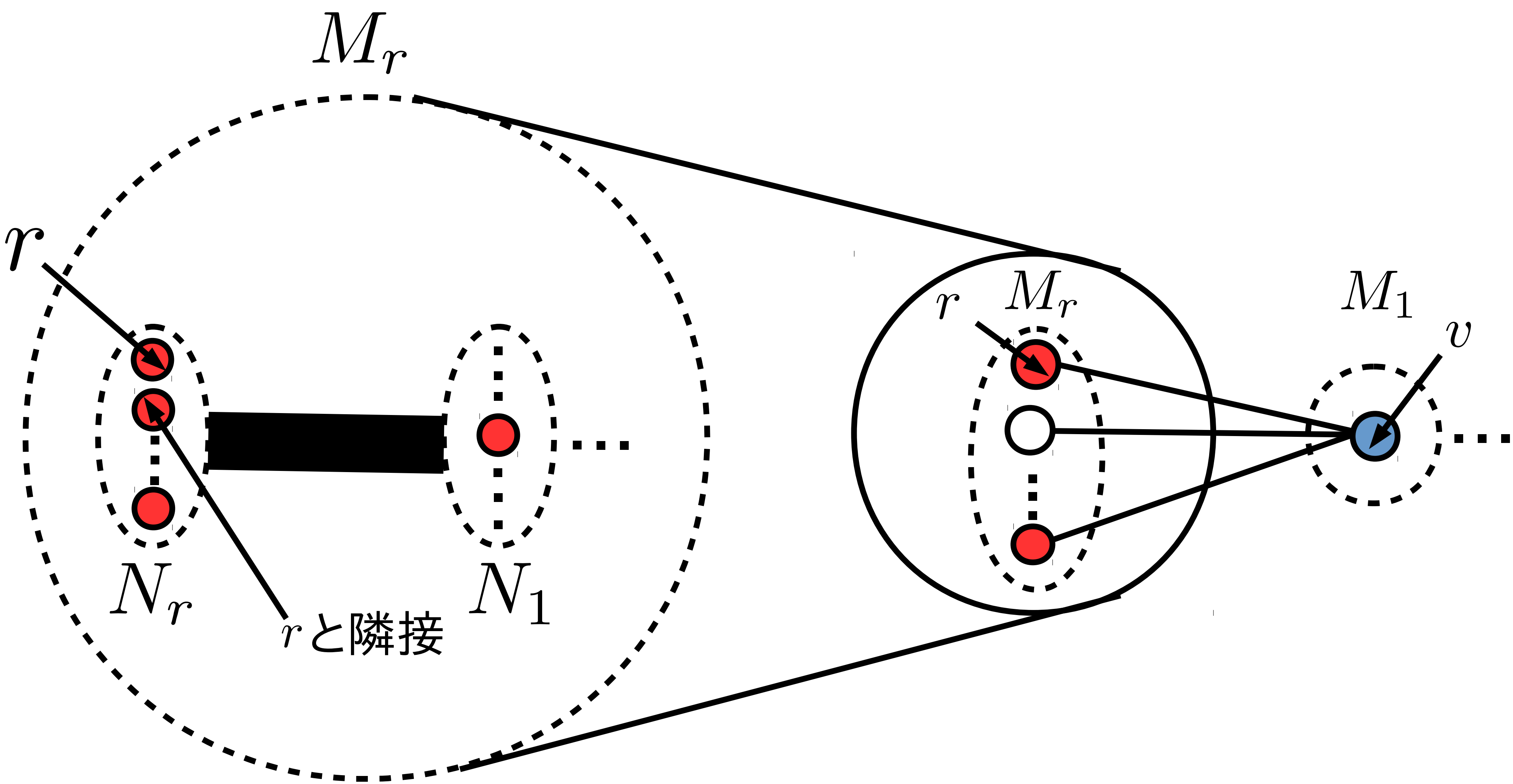
時刻 $t = 0$

rを含むモジュールの観察



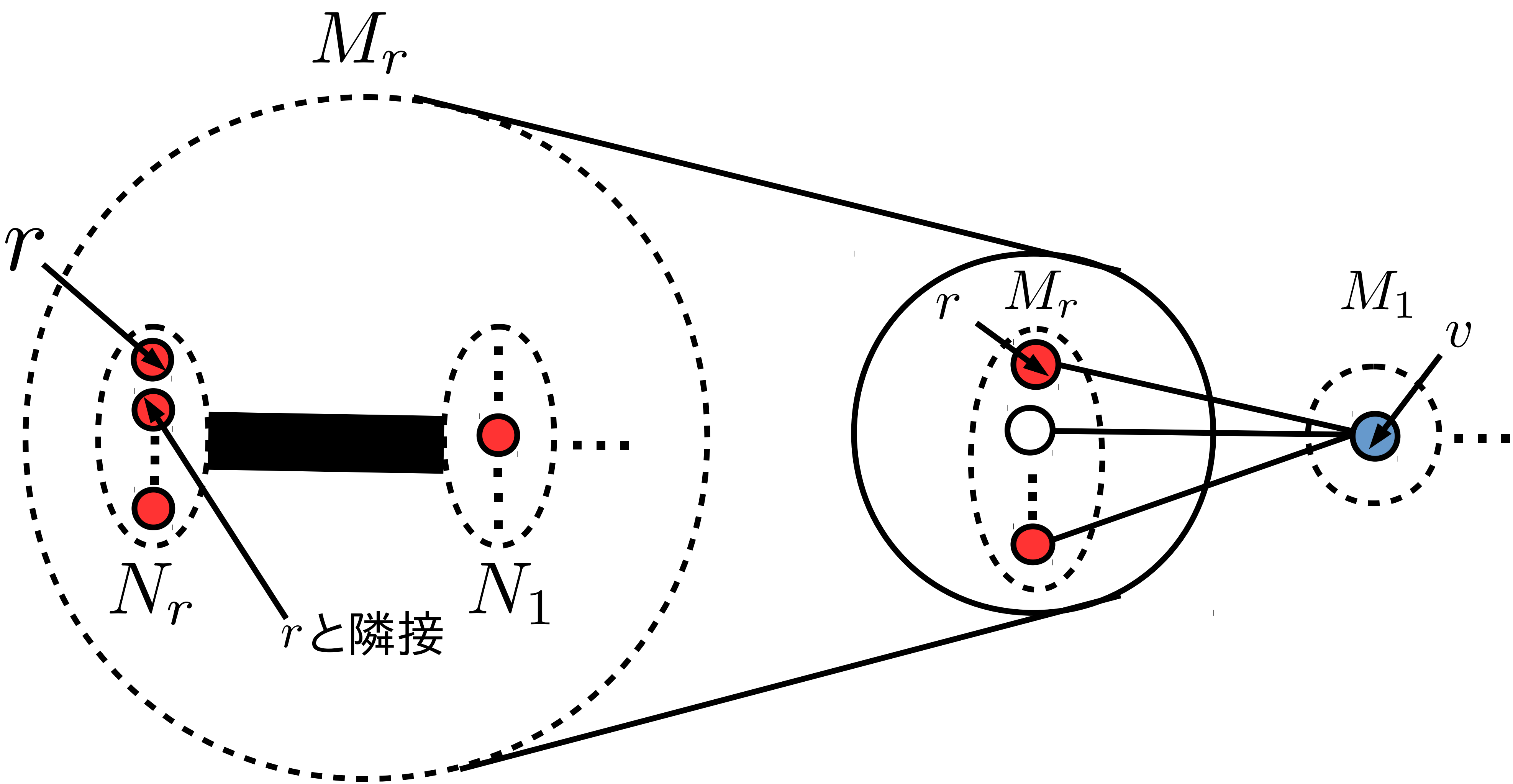
時刻 $t = 1$

rを含むモジュールの観察



時刻 $t = 2$

rを含むモジュールの観察



時刻 $t = 2$

N_r の守られていない全ての頂点に火が点く

場合分け

M_r 以外の頂点に火が点いた場合

1. r が M_r 以外の2つ以上の頂点と隣接しているとき
   戦略の総数　：　k^{k-1}

2. r が M_r 以外の1つの頂点 v と隣接しているとき
   次の2つの場合を実行する.

   2.1　時刻 $t=0$ と $t=1$ の間で v を守らなかったとき
   戦略の総数　：　k^{k-1}

上記以外の場合

   2.2　時刻 $t=0$ と $t=1$ の間で v を守ったとき
   戦略の総数　：　k^{k-2}

Kumamoto University

効率が良い頂点の守り方

- あるモジュールに火が点くとき, 全ての守られていない点に同時に着火する
- あるモジュールの一部しか守らなかった場合, 火の広がり方には影響がない
- 火の広がりを防ぐためにあるモジュールM内の全ての頂点を守るならば, M内の点を全て続けて守ると仮定してよい

Kumamoto University

戦略を求めるアルゴリズム

1. 守るモジュールの集合を選ぶ

2. モジュールを守る順番を決める

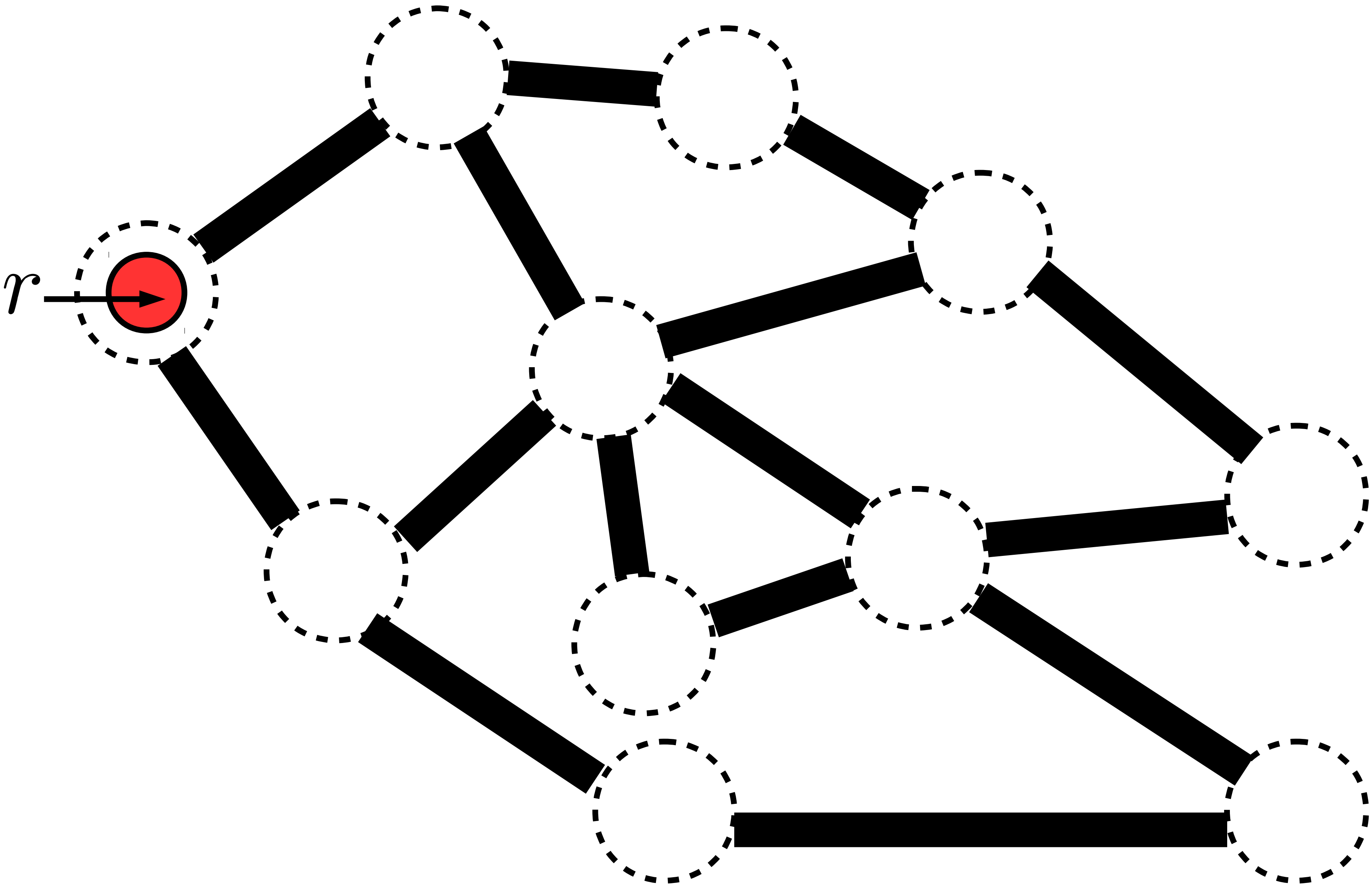
3. 決めた順番で実際に守ってみる

4. 火が広がりきるまで，守れるところを適当に守る

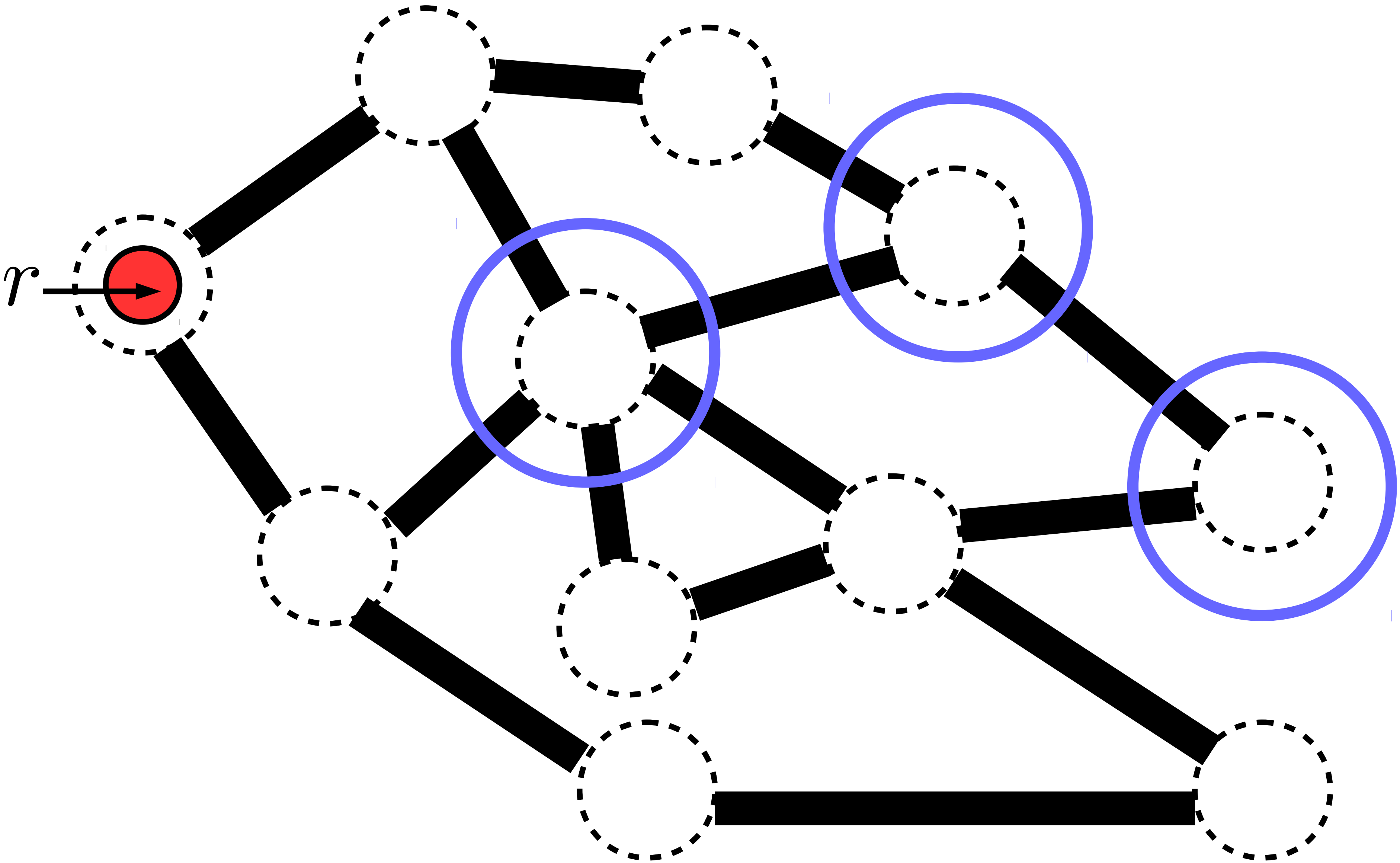
戦略を求めるアルゴリズム

1. 守るモジュールの集合を選ぶ
2. モジュールを守る順番を決める
3. 決めた順番で実際に守ってみる
4. 火が広がりきるまで，守れるところを適当に守る

モジュールの選択例



モジュールの選択例



戦略を求めるアルゴリズム

1. 守るモジュールの集合を選ぶ
2. モジュールを守る順番を決める
3. 決めた順番で実際に守ってみる
4. 火が広がりきるまで，守れるところを適当に守る

Kumamoto University

モジュールを守る順番

モジュール $M_1, \ldots, M_n$ を守る順番の決め方

a) G 上で r に最も近いモジュール M_i を選択する.

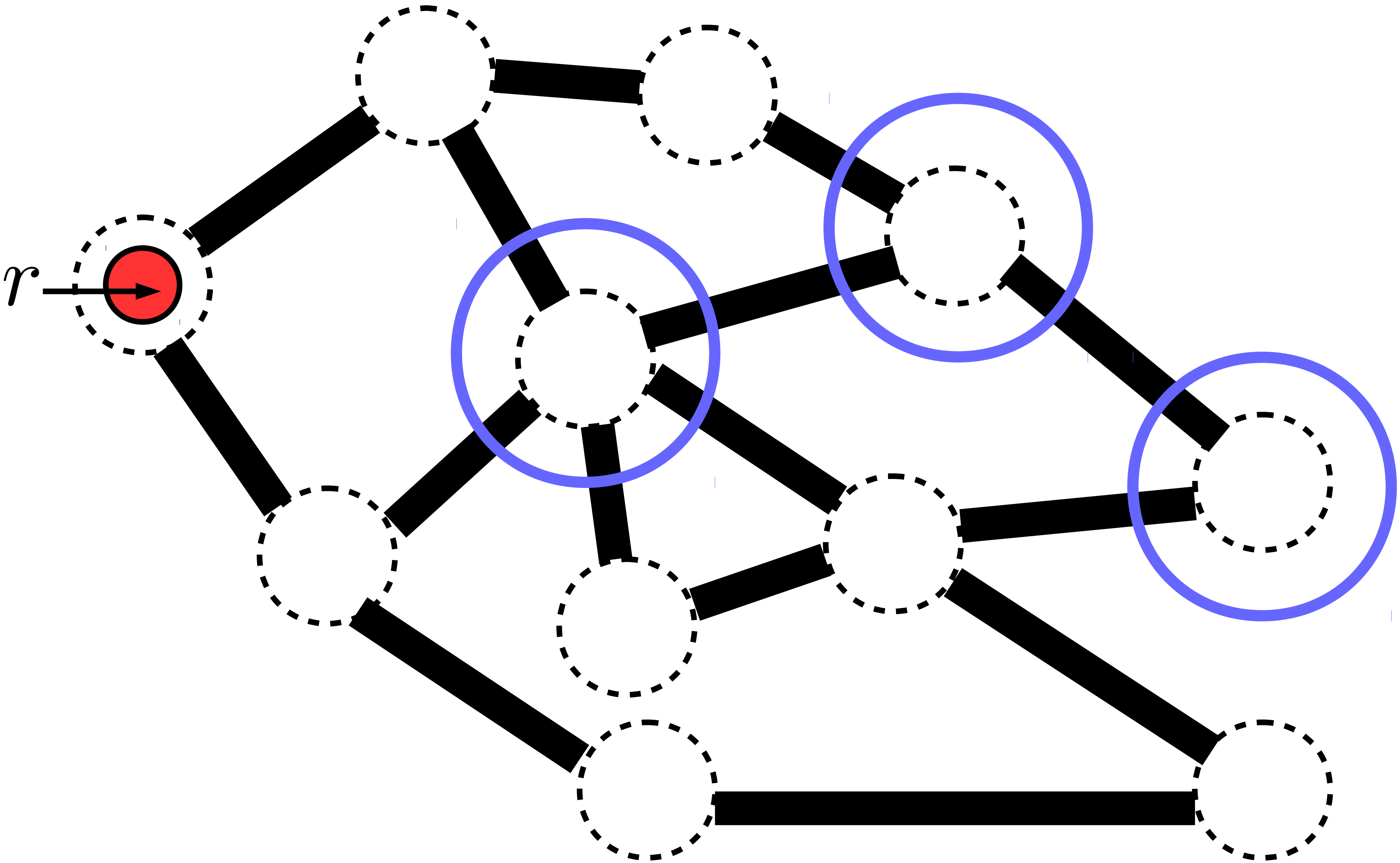
b) G をグラフ $G - M_i$ に置き換える

c)全てのモジュールを選択するまで上記 a) , b) を繰り返す

d)選択したモジュール順に守ることとする

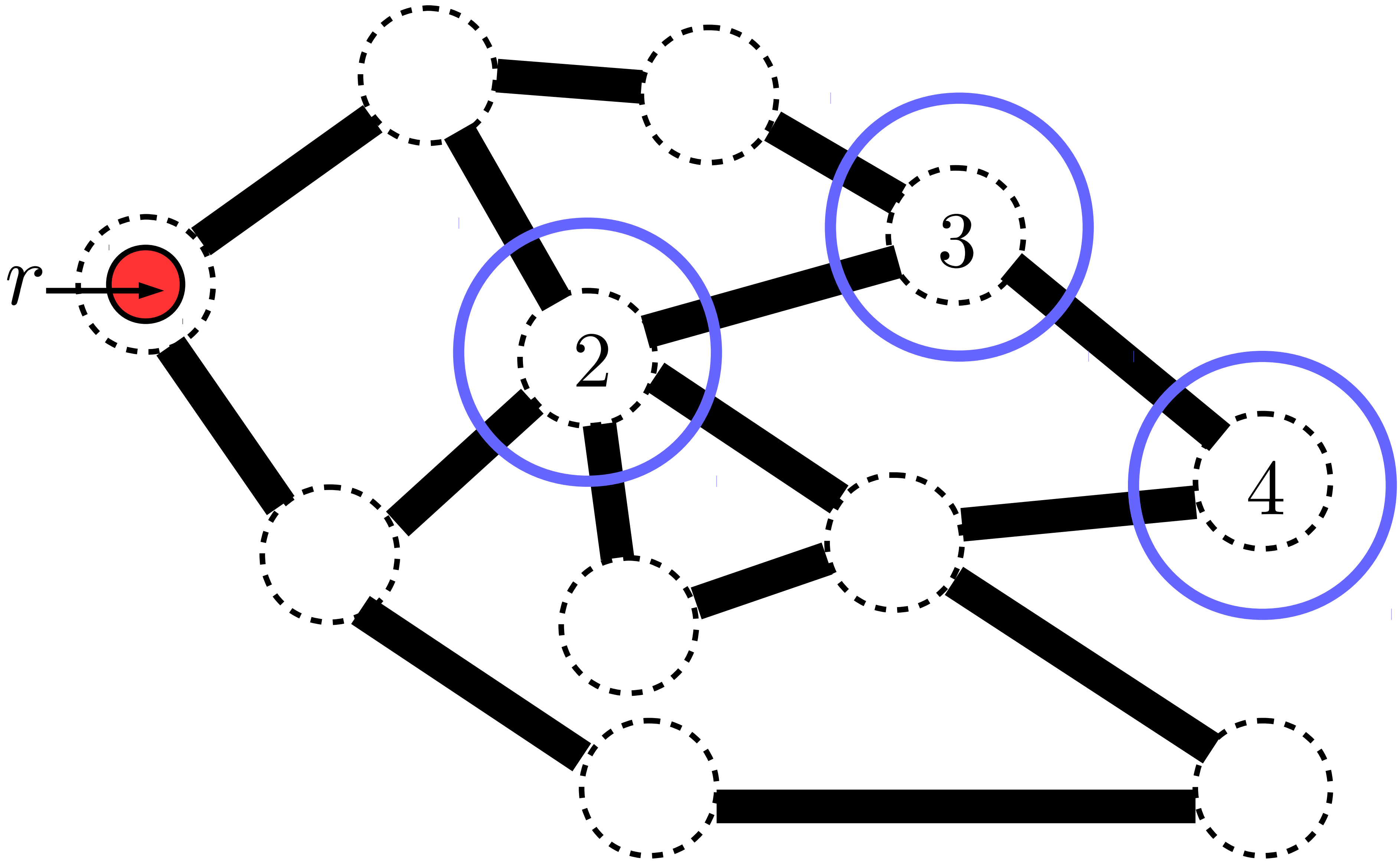
Kumamoto University

モジュールの順番例



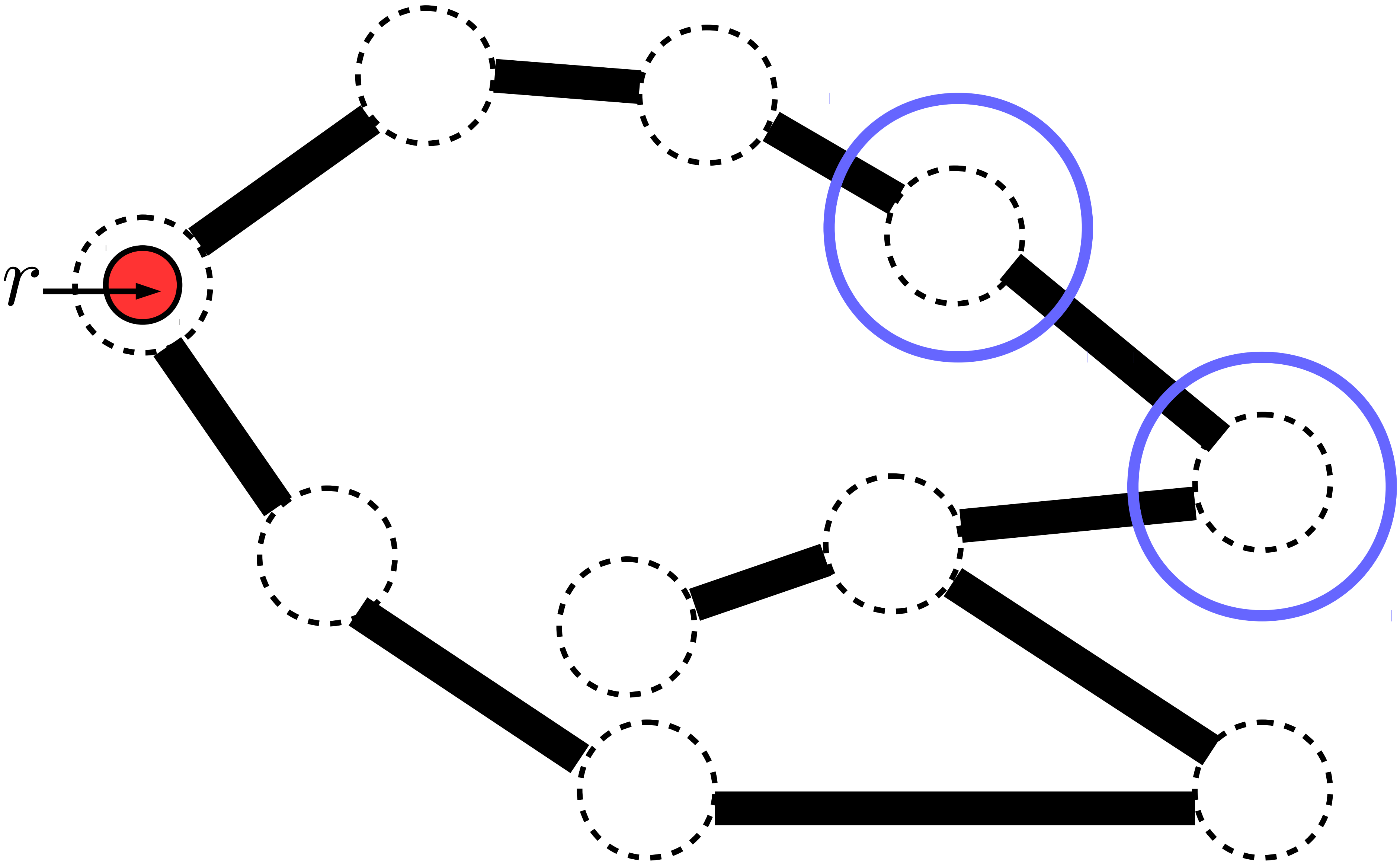
Kumamoto University

モジュールの順番例

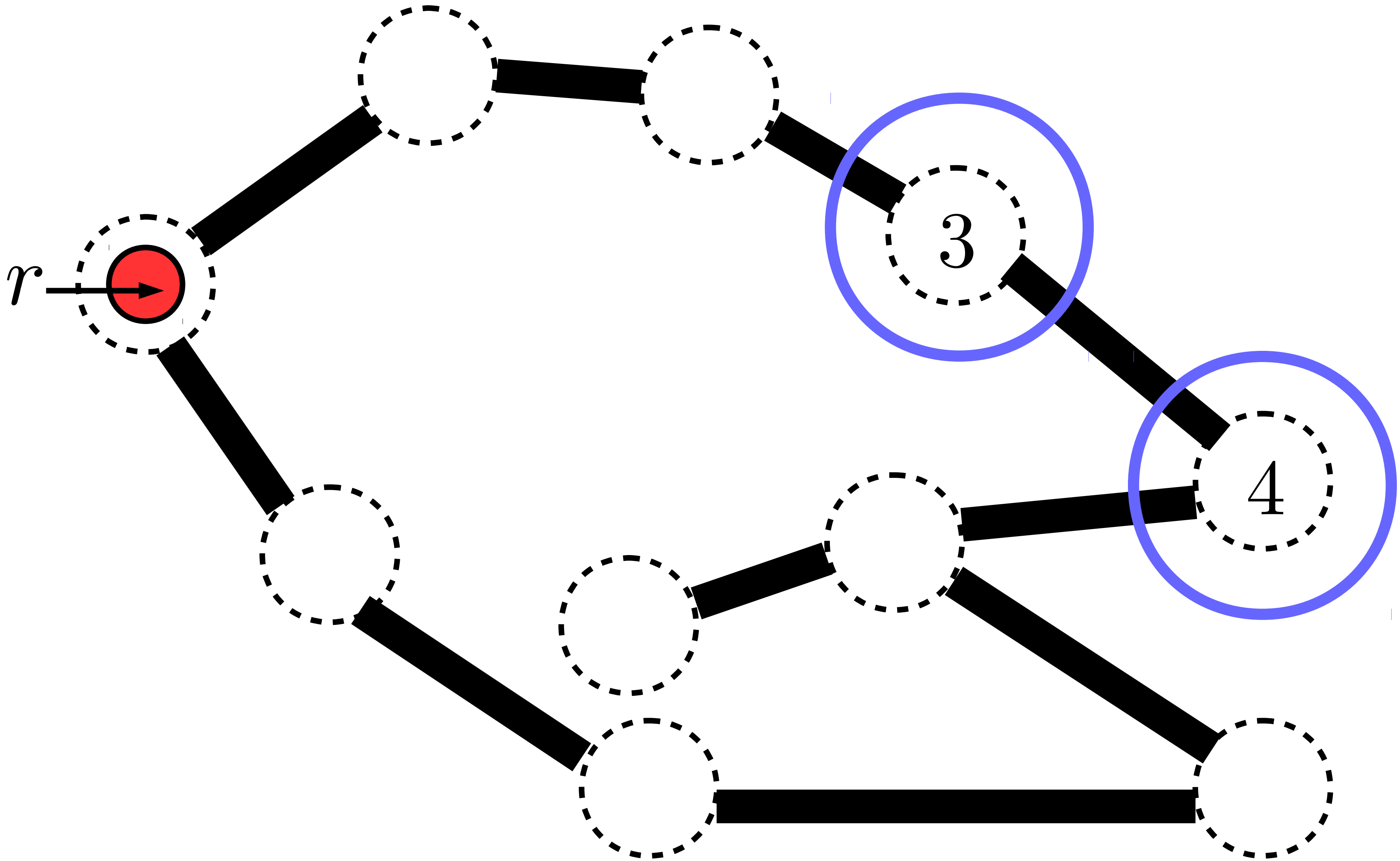


Kumamoto University

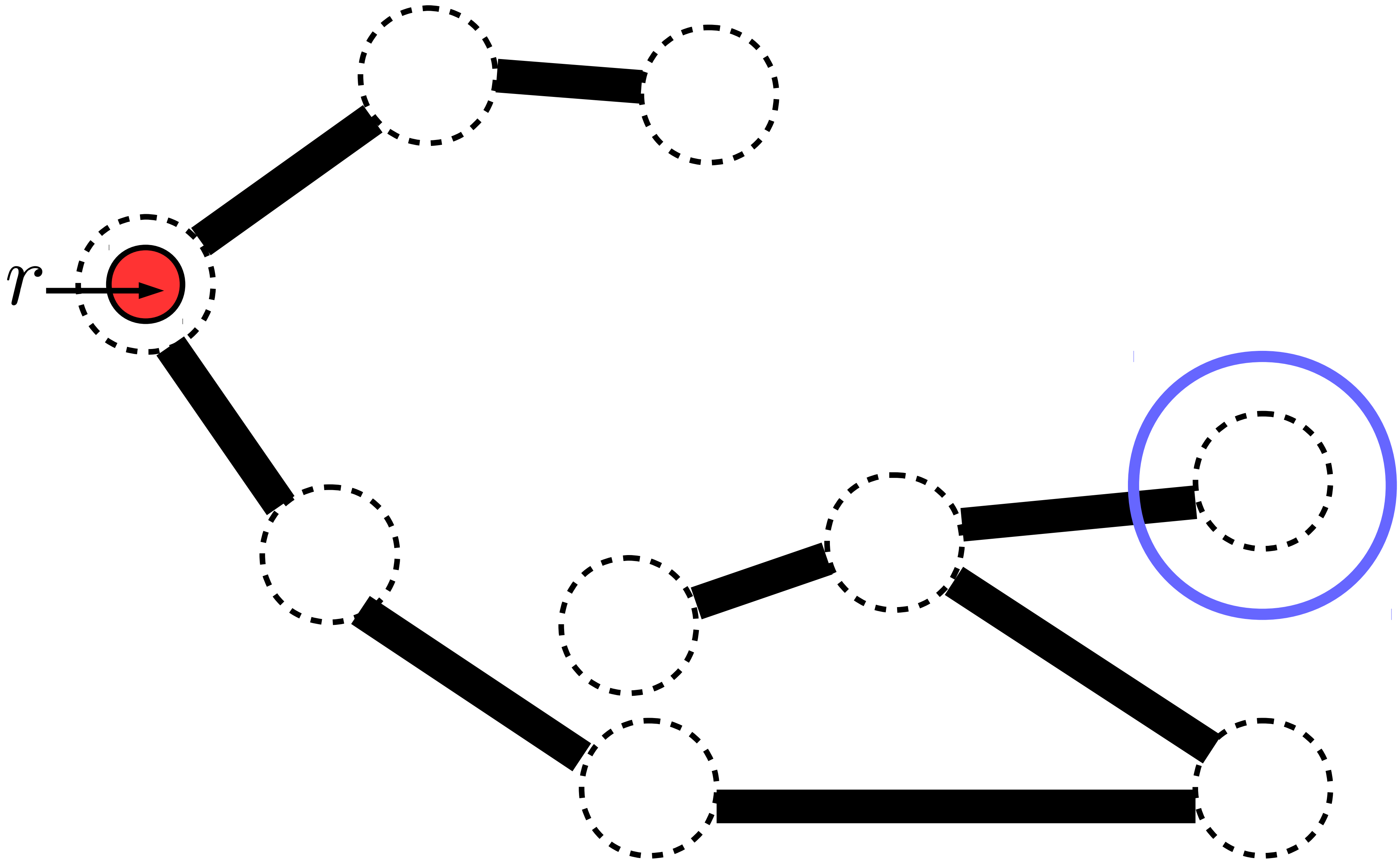
モジュールの順番例



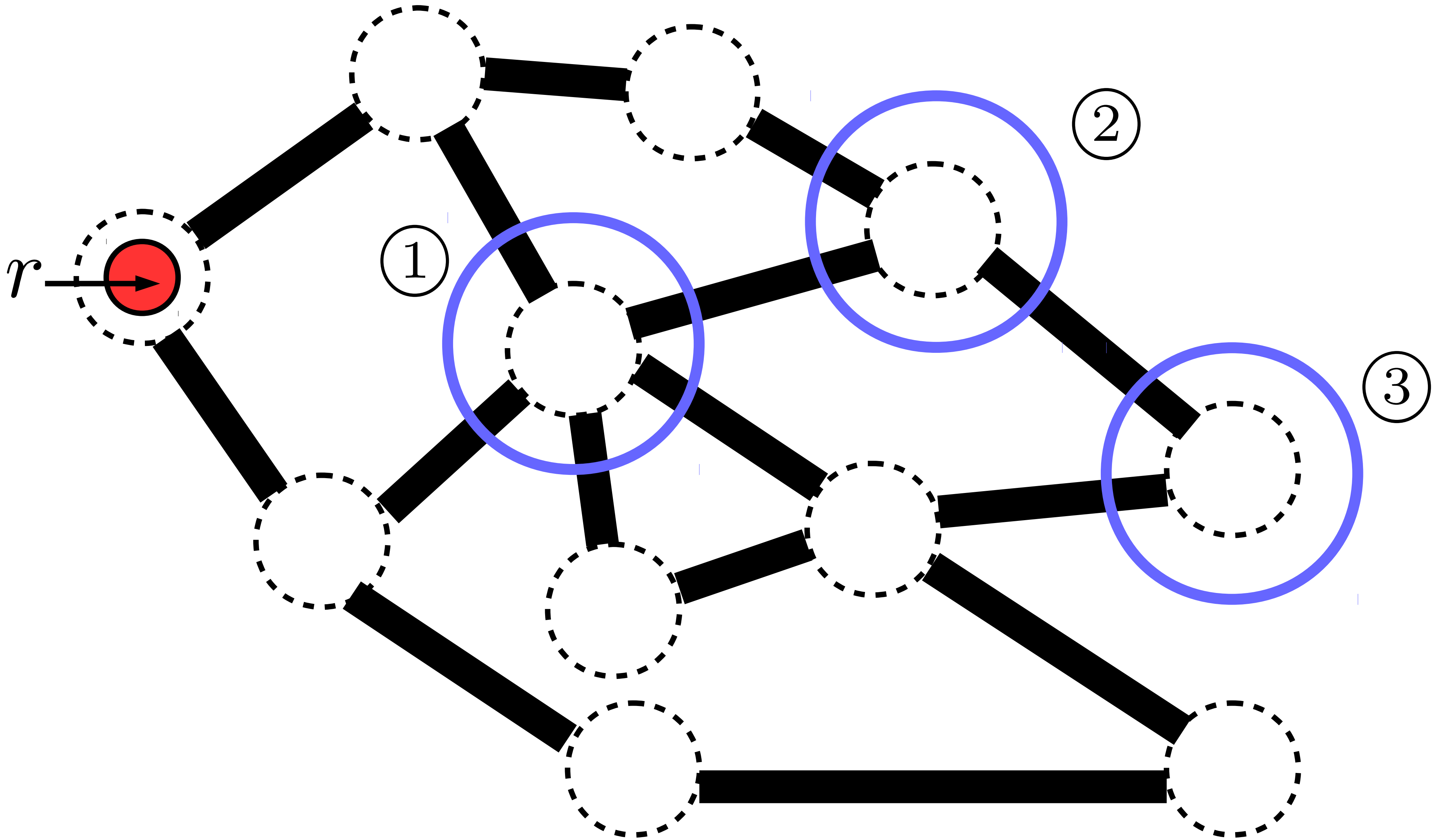
モジュールの順番例



モジュールの順番例



モジュールの順番例



戦略を求めるアルゴリズム

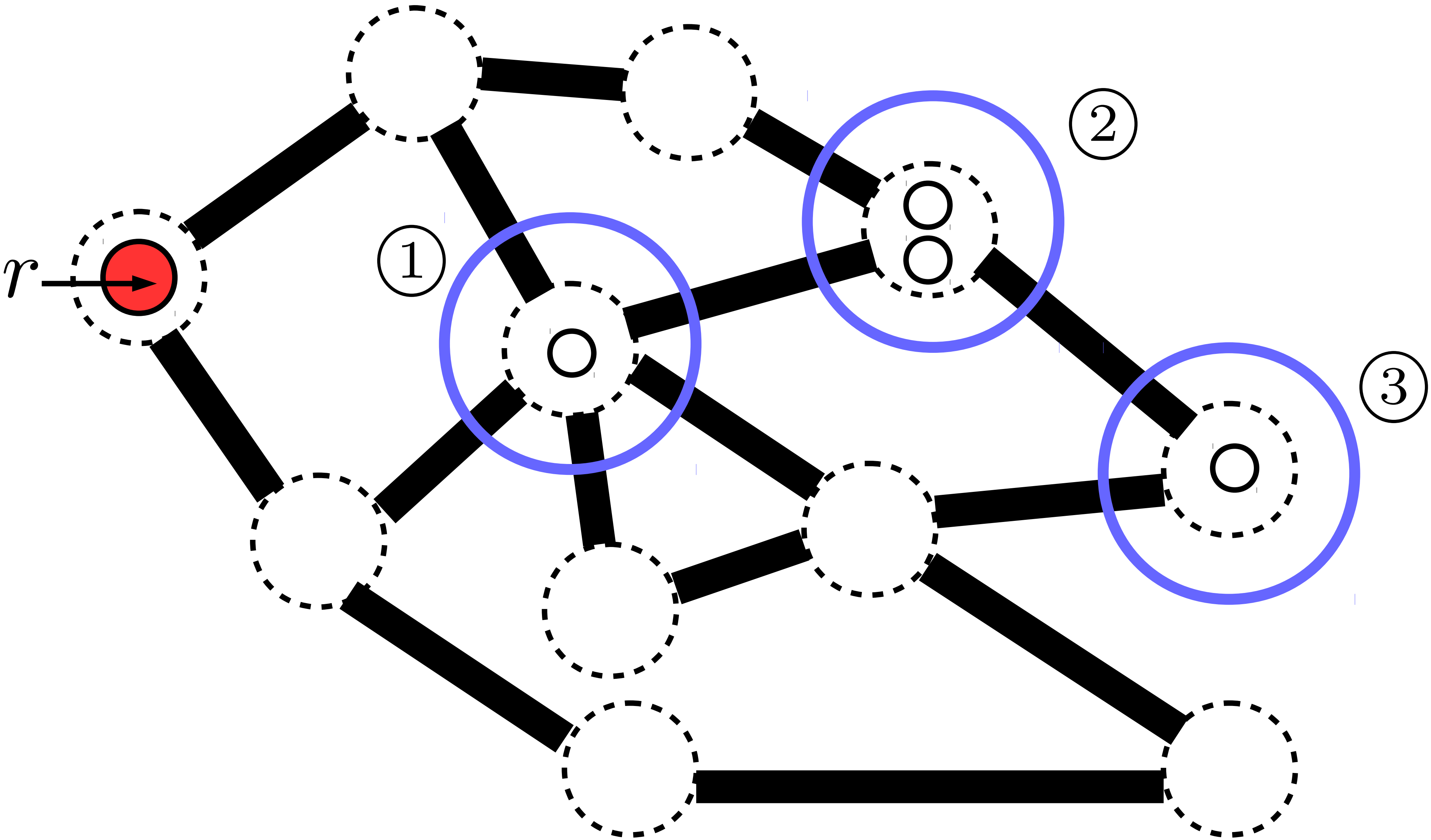
1. 守るモジュールの集合を選ぶ

2. モジュールを守る順番を決める

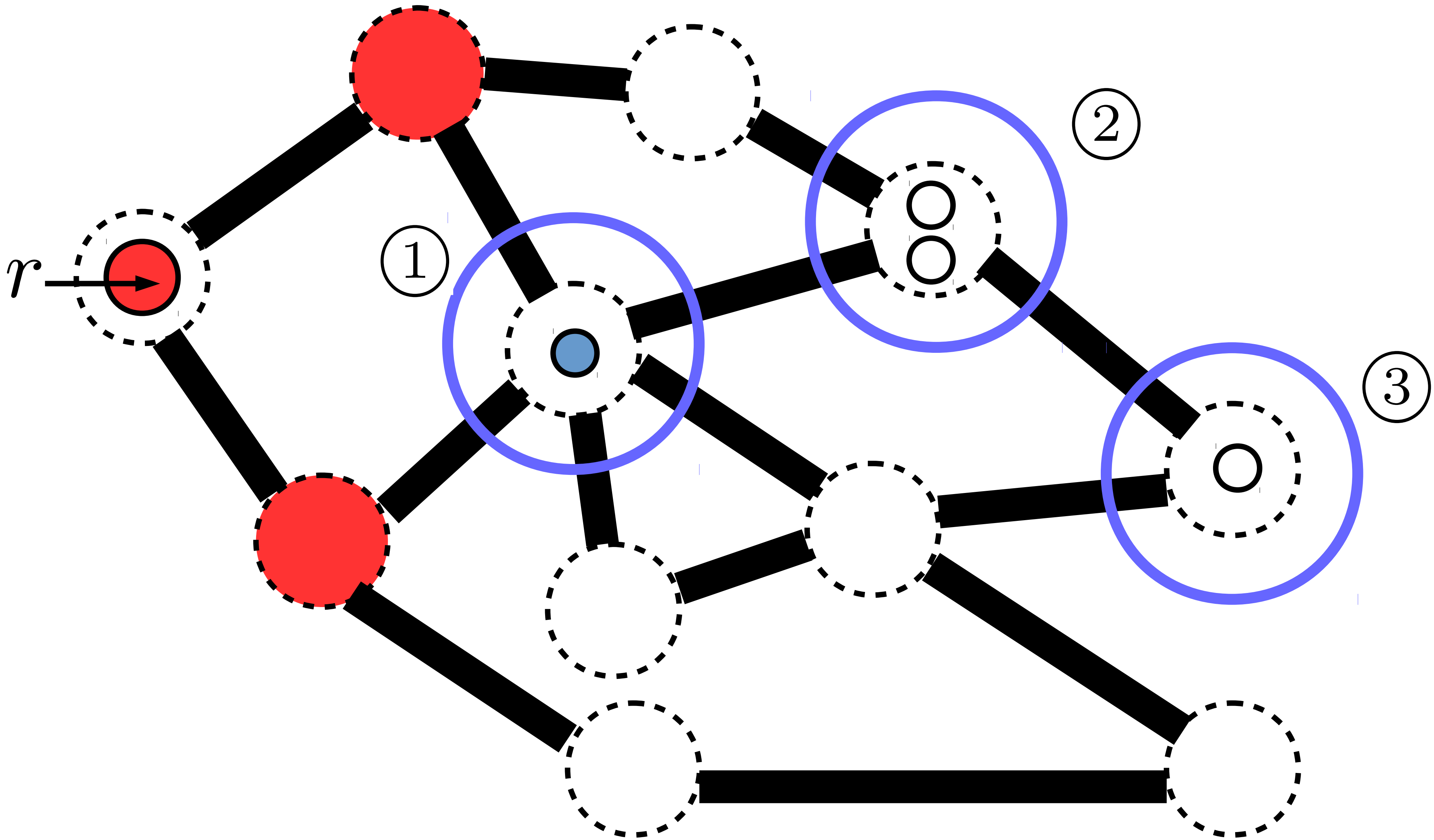
3. 決めた順番で実際に守ってみる

4. 火が広がりきるまで，守れるところを適当に守る

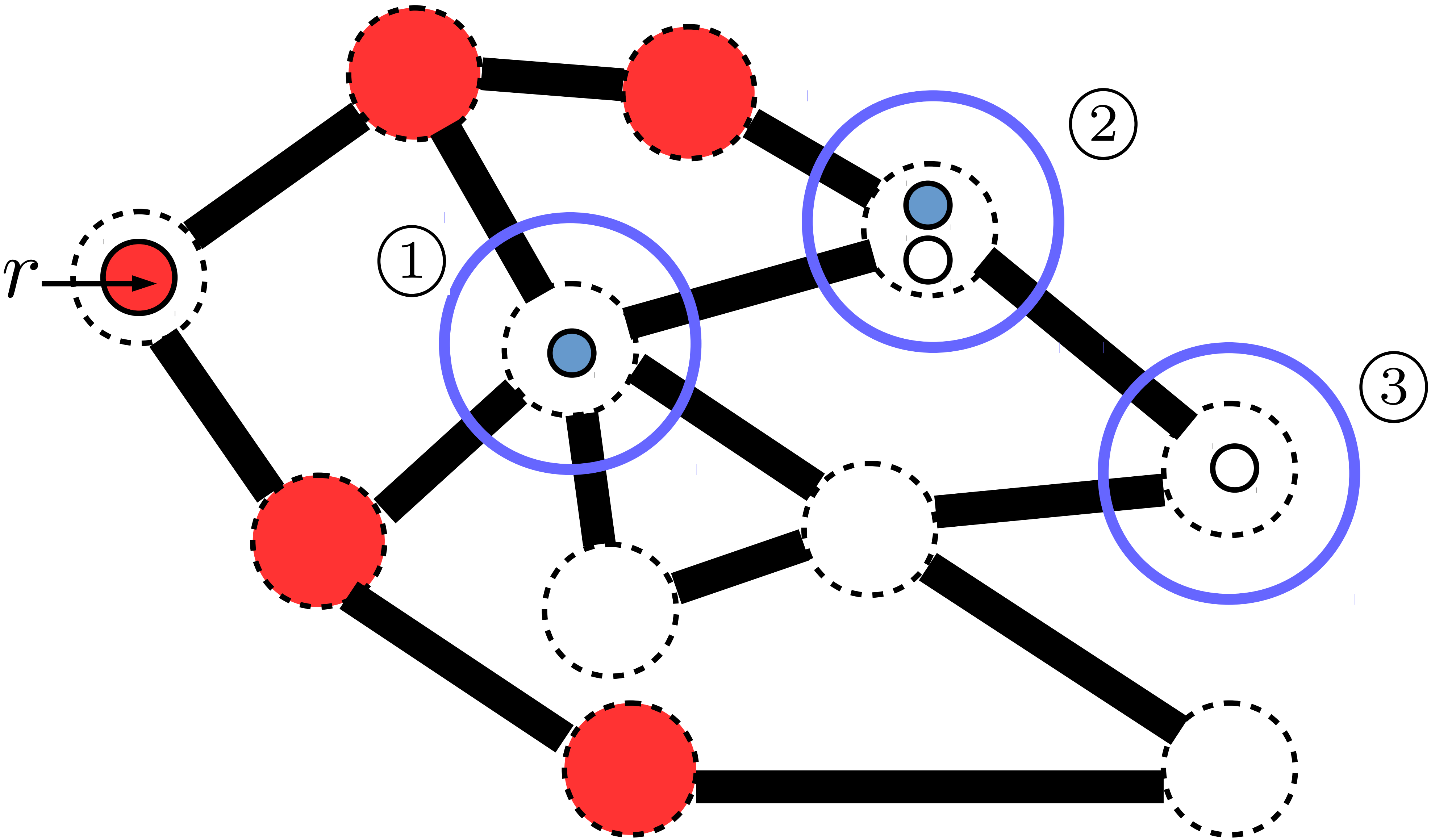
モジュールを順に守る例



モジュールを順に守る例

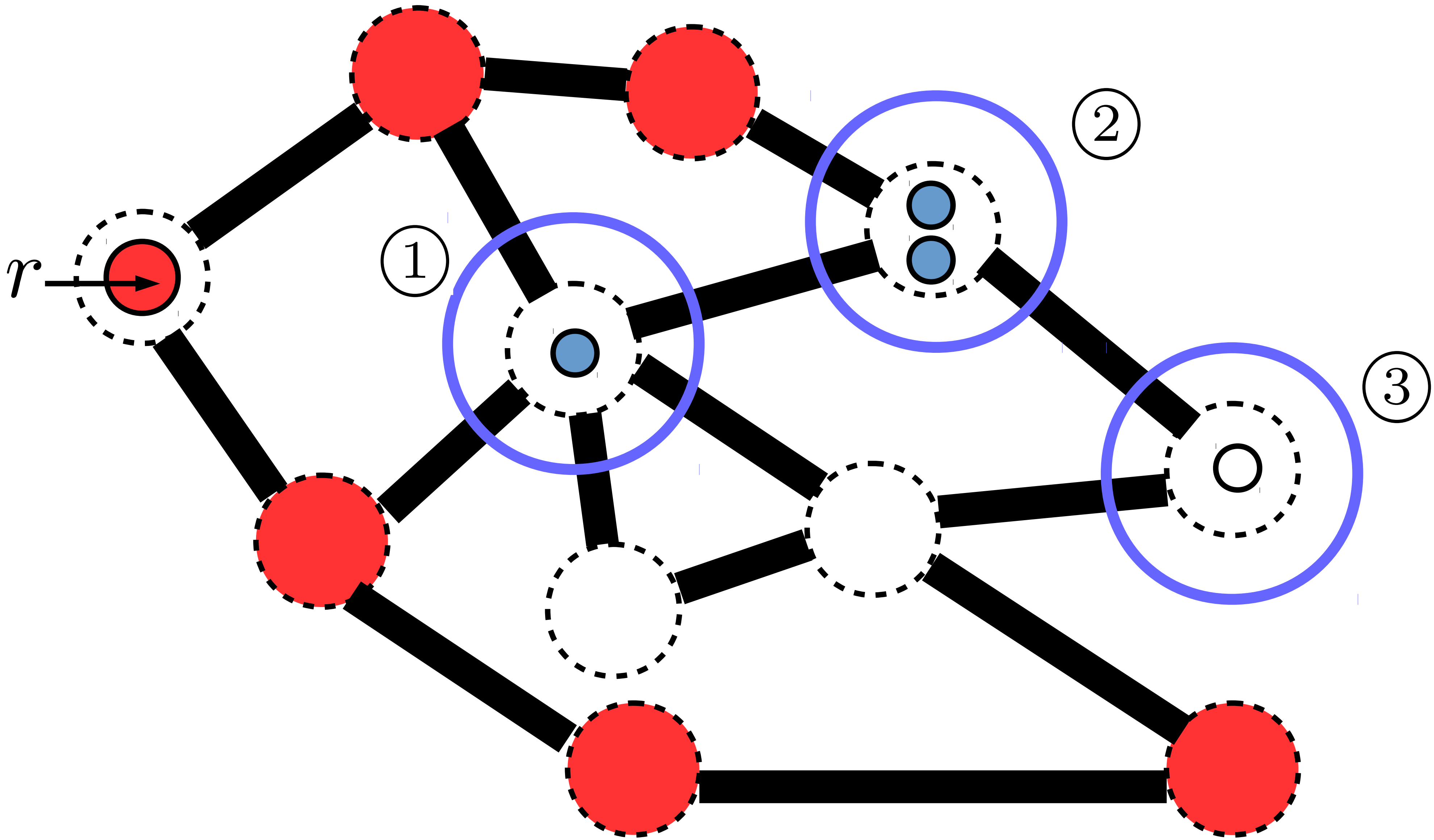


モジュールを順に守る例

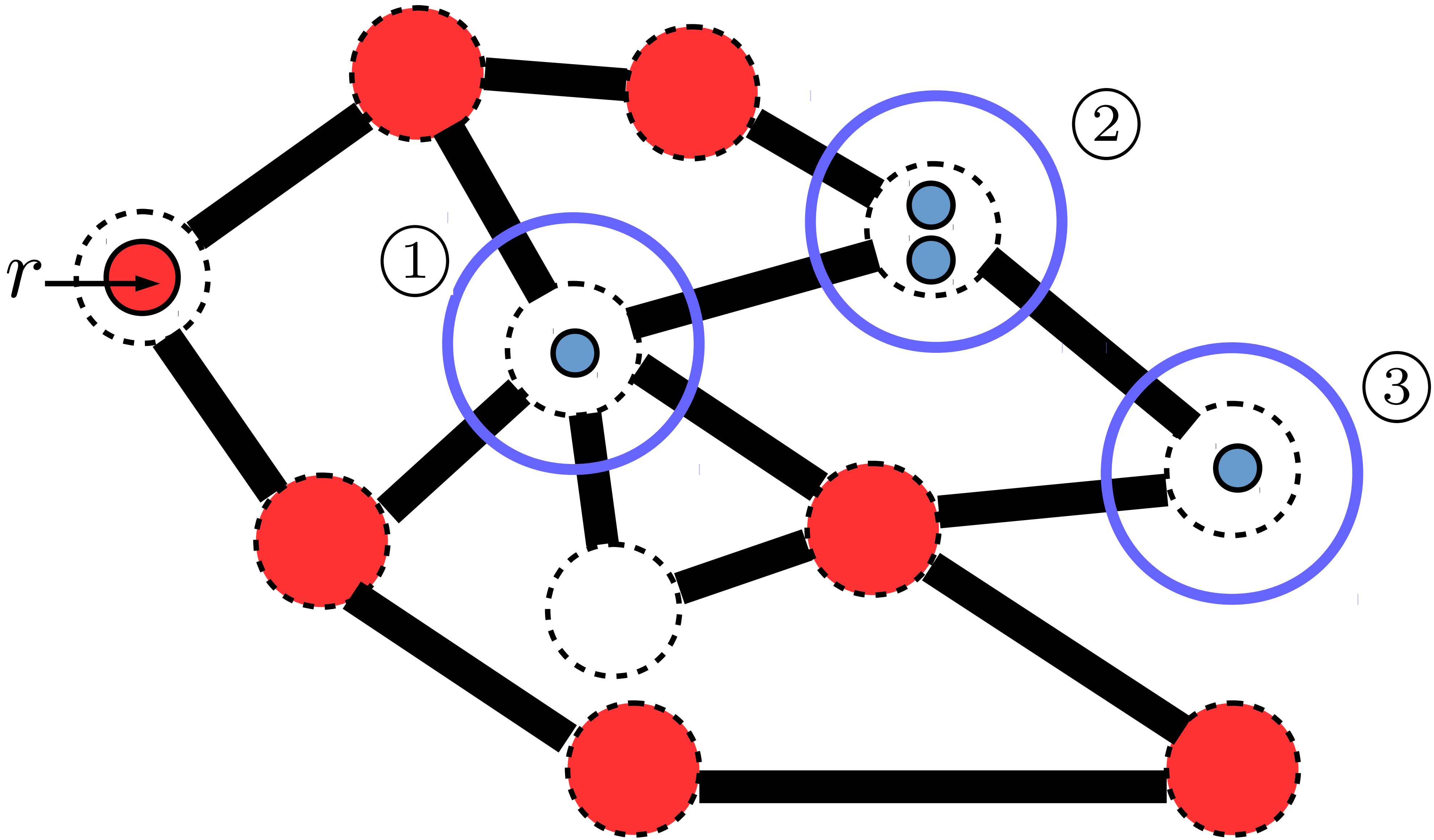


Kumamoto University

モジュールを順に守る例



モジュールを順に守る例



Kumamoto University

戦略を求めるアルゴリズム

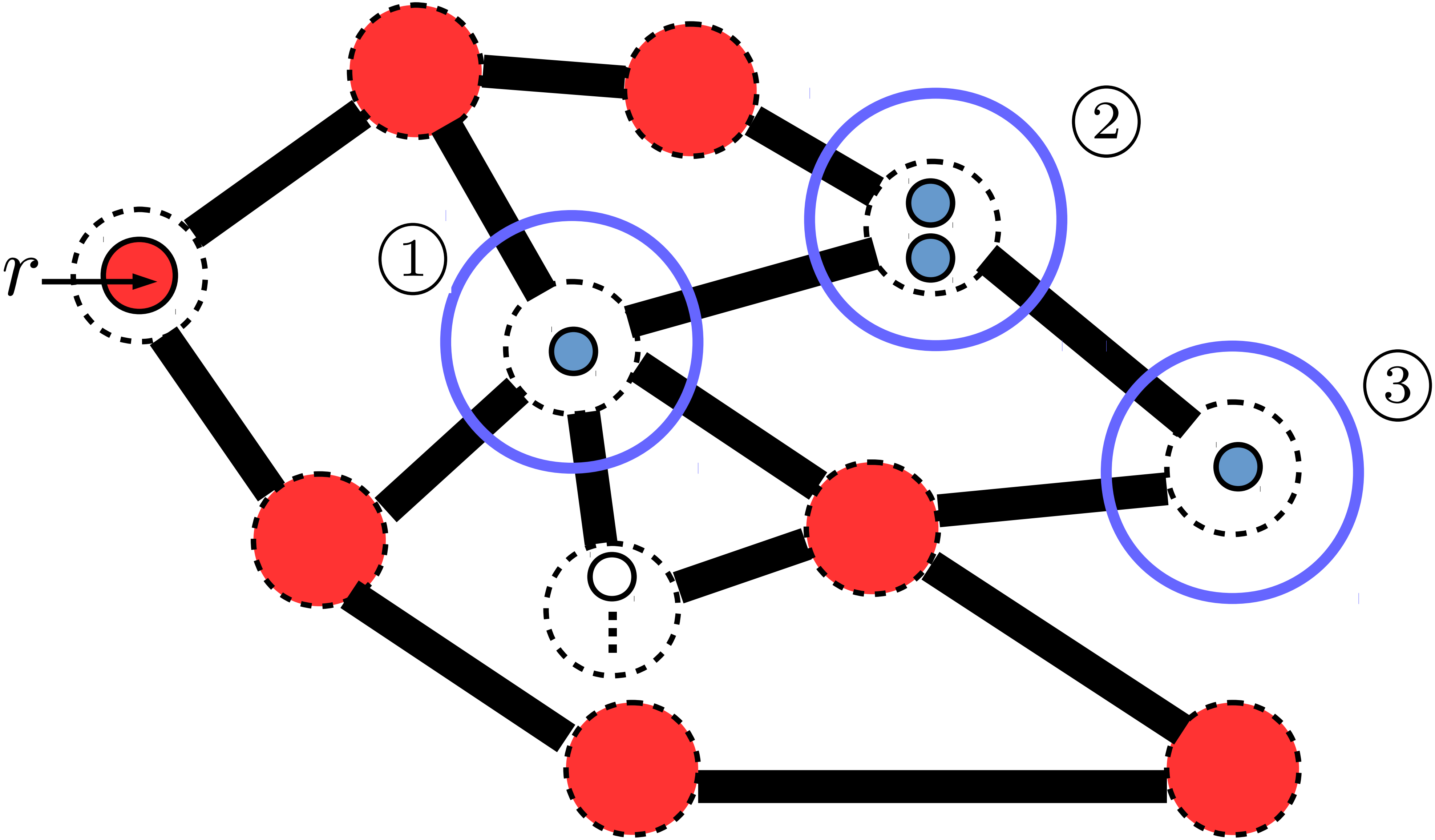
1. 守るモジュールの集合を選ぶ

2. モジュールを守る順番を決める

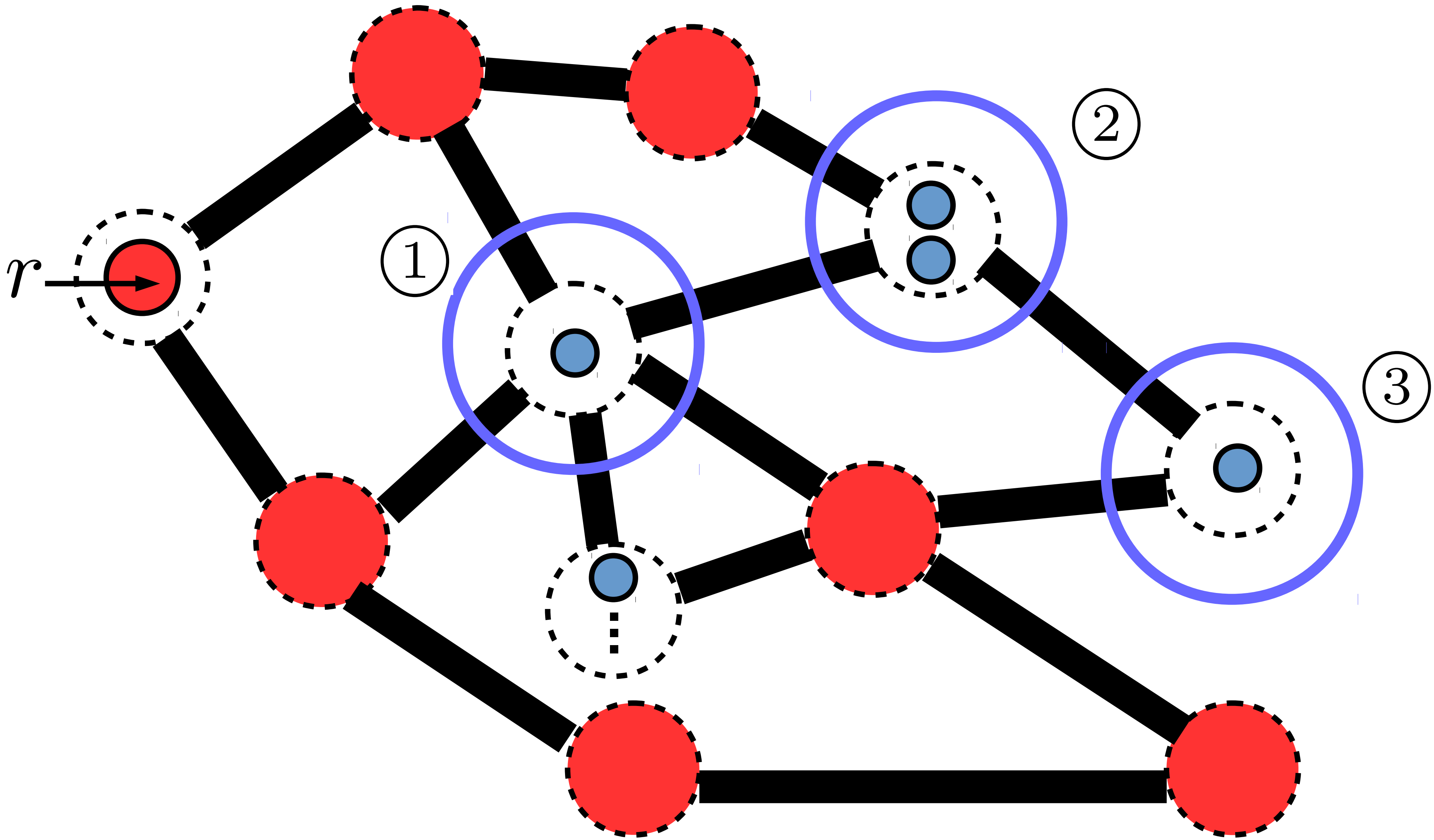
3. 決めた順番で実際に守ってみる

4. 火が広がりきるまで, 守れるところを適当に守る

守れる頂点を適当に守る例



守れる頂点を適当に守る例



命題2(再掲)

1. 守るモジュールの集合を選ぶ
2. モジュールを守る順番を決める
3. 決めた順番で実際に守ってみる
4. 火が広がりきるまで, 守れるところを適当に守る

命題2 [Fomin, Heggernes, van Leeuwen. 2016.]

$G=(V,E)$ を入力したFirefighting problemを考える. ある戦略が有効であるかどうかを確認し, かつ有効であればその戦略の解を求めることを $O(|E|+|V|)$ 時間で行うことができる.

計算量

- 全ての戦略を求める計算量

rから最も近い
モジュールを求める回数

$$O(\underbrace{2^k}_{\text{モジュールの組合せの総数}} \cdot \overbrace{k}^{r\text{から最も近いモジュールを求める回数}} \cdot \underbrace{(|E| + |V|)}_{r\text{から最も近いモジュールを求める計算量}})$$

モジュールの
組合せの総数

rから最も近いモジュールを
求める計算量

※ kはmodular width

- 全ての戦略を実行して解を求める計算量

$$O(\underbrace{2^k}_{\text{戦略の総数}} \cdot \underbrace{(|E| + |V|)}_{\text{命題2のアルゴリズムの計算量}})$$

戦略の総数　命題2のアルゴリズムの計算量

Kumamoto University

まとめ

- 結果

modular widthが k であるグラフ $G=(V,E)$を入力としたFirefighting problemは $O(2^k \cdot k \cdot (|E|+|V|))$ 時間で解くことができる.

- 今後の課題

各時刻に複数頂点守れる一般化問題