G: Derangement
- 完全順列 -

原案：井上

解答：井上

問題概要

- 順列pが与えられるので、完全順列となるように並べ替えたい
  - 順列qが完全順列 ⇔ $q_i \neq i$ for all i
- 操作は任意の2要素p_i, p_jの交換のみで、1回のコストは$(p_i+p_j)\times|i-j|$
- コストを最小化せよ

完全順列

例： **12354** → **21354** → **23154**

(1+2)×1　(1+3)×1

コスト7

想定解法: 最小費用流

正確には
最小重み完全2部
マッチング

補題1: 順列pを特定の順列qに並べ替えるための最小コストは、

1. p_i, p_jの交換をコスト$(p_i+p_j)\times|i-j|$で行う操作
2. p_iを位置jにコスト$p_i\times|i-j|$で割り当てる操作

のどちらでも等しくなる

52314 $\xrightarrow{(5+3)\times|1-3|}$ 32514

52314 $\xrightarrow{4\times|5-1|}$ 42513

補題2：p_iを位置jにコスト$p_i\times|i-j|$で割り当てる操作を用いて
順列pを完全順列に並べ替えるための最小コストは、
最小費用流を用いて求めることができる

想定解法: 最小費用流

定理: p_i, p_jの交換をコスト$(p_i+p_j) \times |i-j|$で行う操作を用いて順列pを完全順列に並べ替えるための最小コストは、最小費用流を用いて求めることができる

証明：補題2により求めた最小コストをC、コストCを実現する順列をqとする。補題1より、順列qは入替操作によってもコストCで得られる。入替操作によりコストC未満で得られる完全順列q′を仮定すると、q′は割当操作によってもC未満で得られることになり、Cが最小コストであることに矛盾する。■

補題1: 入替最小=割当最小

- 割当コスト$p_i \times |i-j|$は入替コスト$(p_i+p_j) \times |i-j|$を分解したものになっているので、入替最小≧割当最小

$$\mathbf{52314} \xrightarrow{(5+3)\times|1-3|} \mathbf{32514}$$

52314

5×|1-3| 3×|1-3|

32514

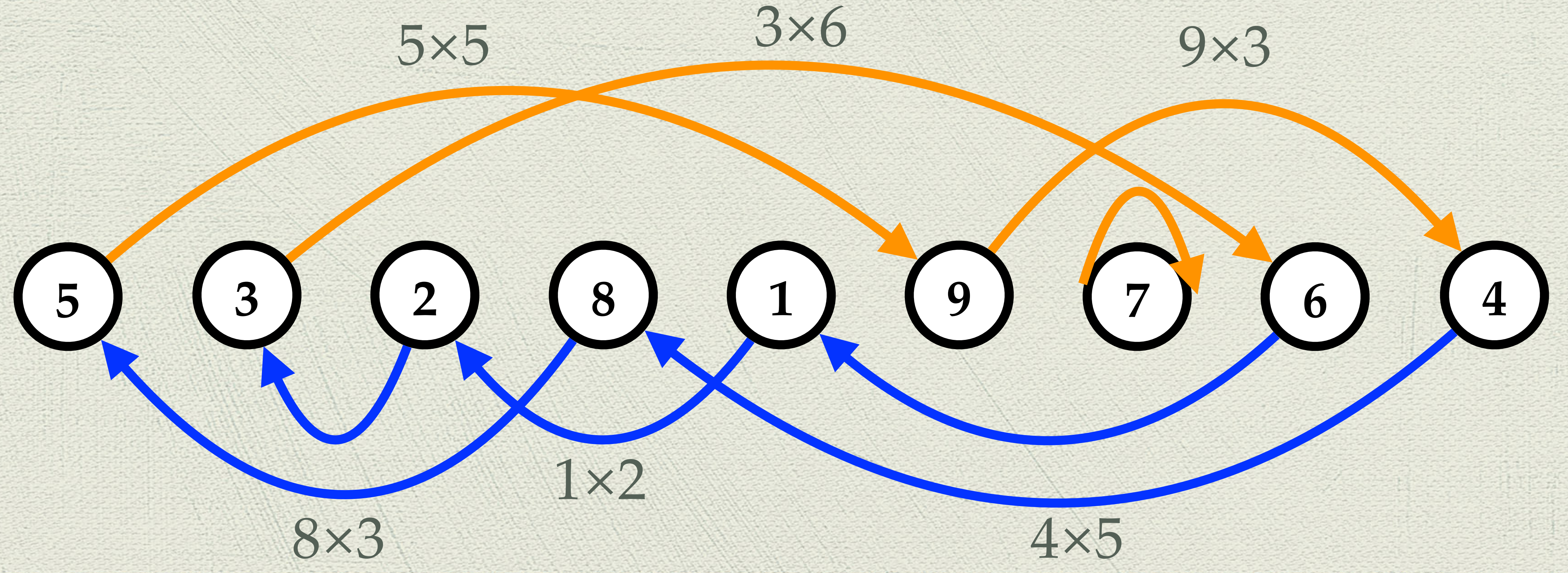
- pからqへの割当最小は、$p_i=q_j$となるjに割り当てること

- よって、入替操作も上記割当最小と等しいコストで可能なことを示せばよい

補題1: 入替最小=割当最小

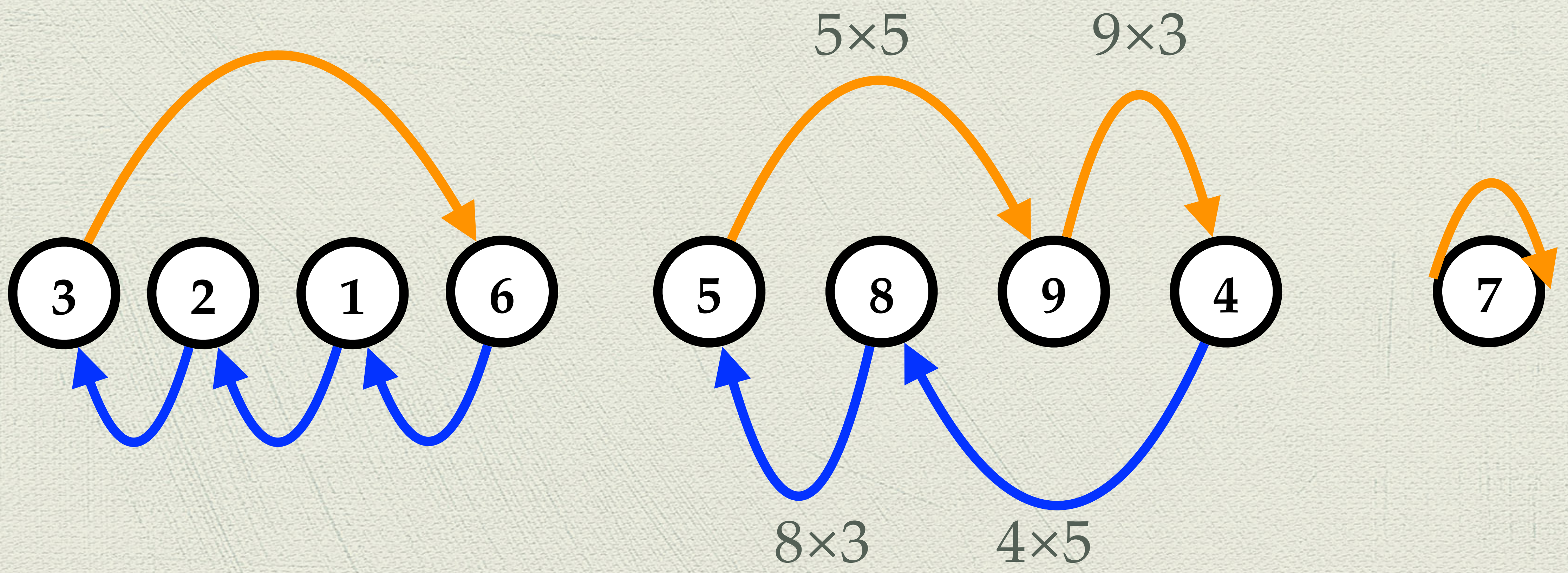
- 割当で行きたいところへ行きたいところへコスト付きの辺を張ったグラフを考える

例: p = 532819764, q = 821465739 (例えば4は4番目に行きたい)



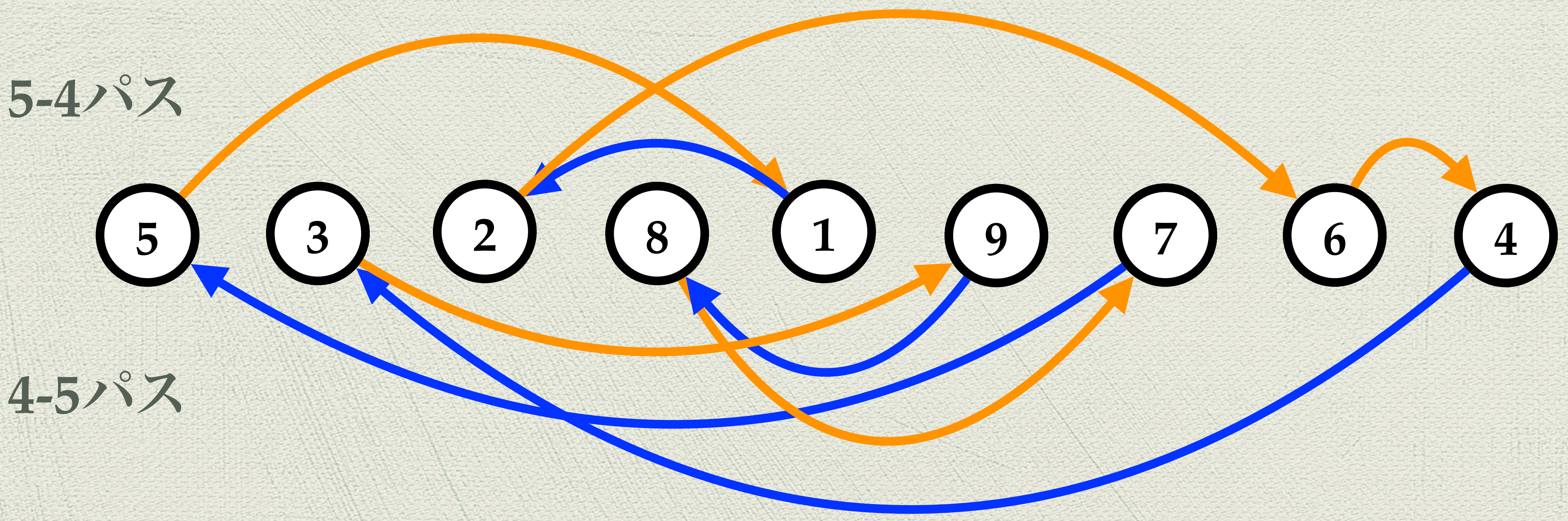
補題1: 入替最小=割当最小

- 割当で行きたいところへ行きたいところへコスト付きの辺を張ったグラフを考える
- サイクルに分解できる。分けて考えてよい



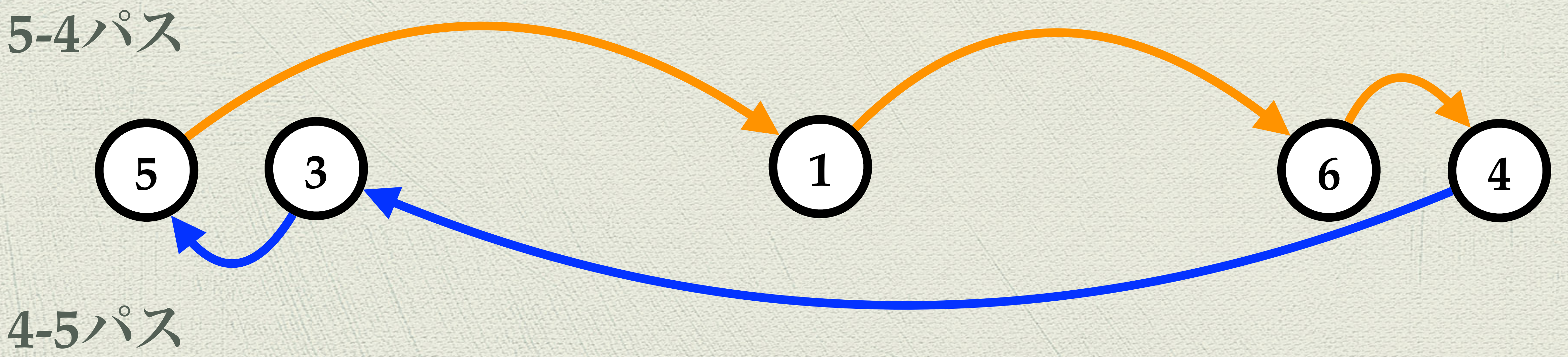
補題1: 入替最小=割当最小

- サイクルになっている = 左端から右端へのパスがある



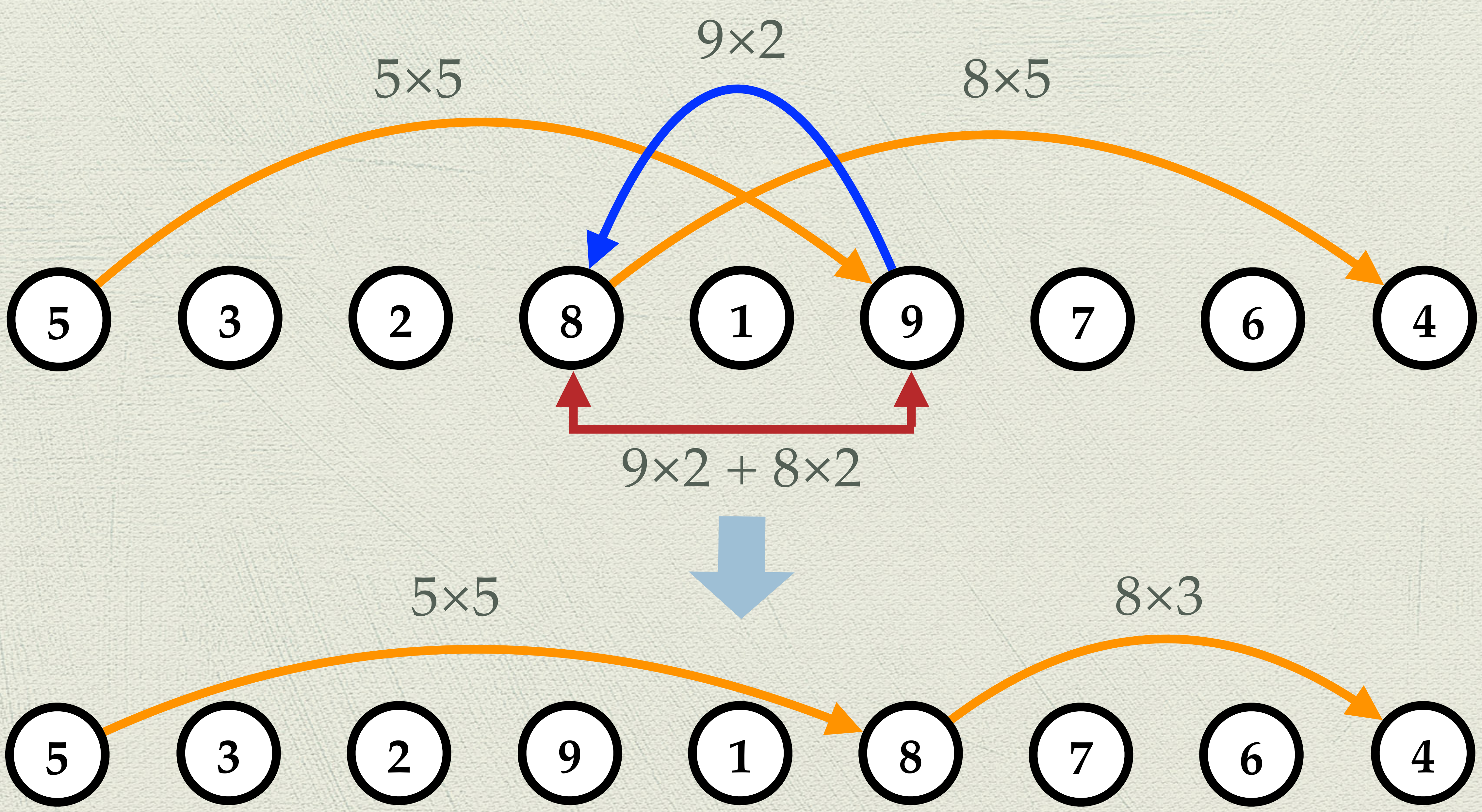
補題1: 入替最小=割当最小

- サイクルになっている = 左端から右端へのパスがある
- これを無駄なコストなしの操作で順路のみにしたい
- 上・下分けて考える



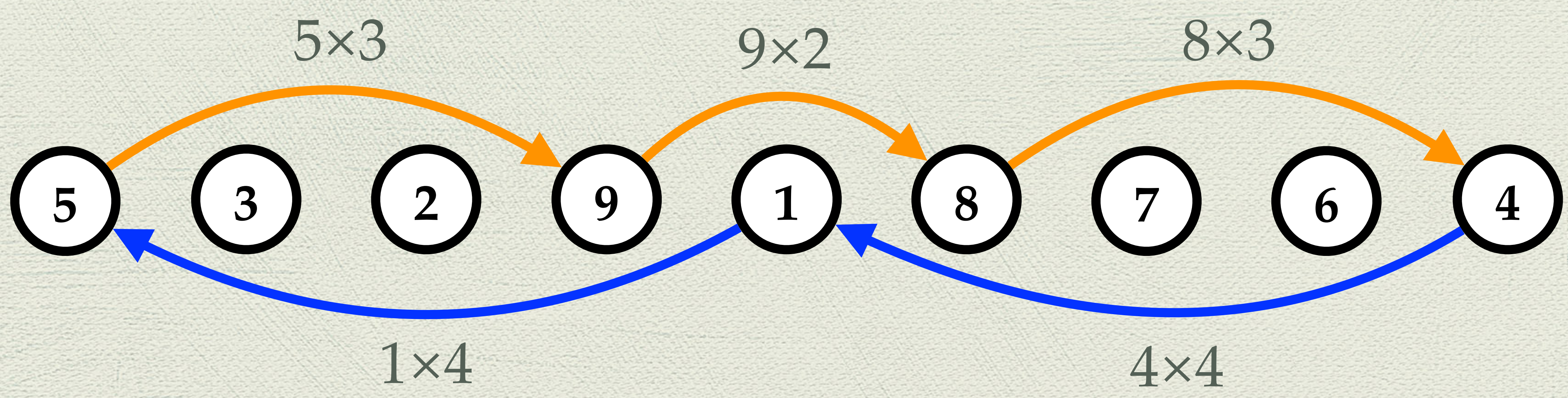
補題1: 入替最小=割当最小

- 逆辺をその行き先と入れ替えればよい



補題1: 入替最小=割当最小

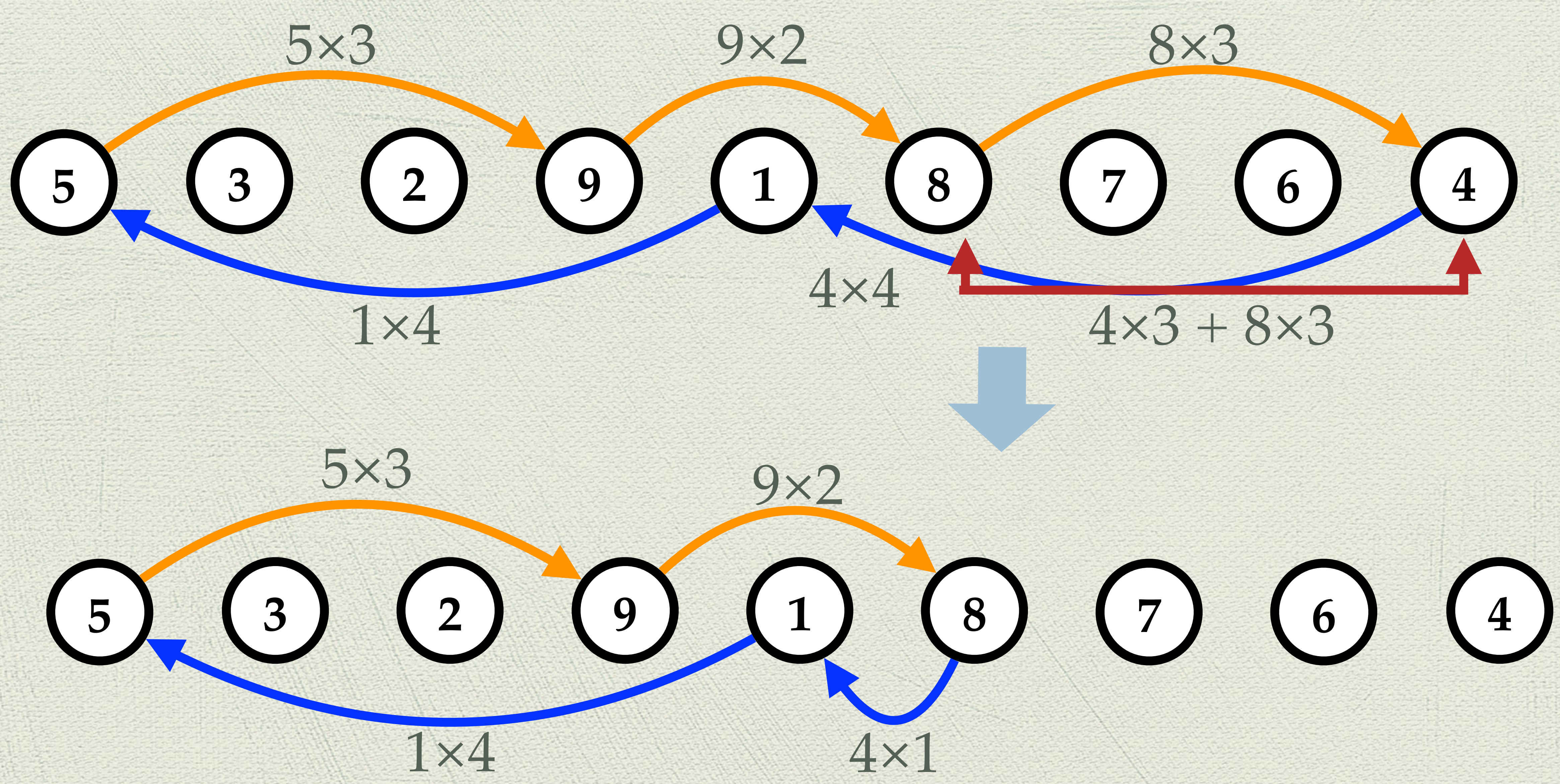
- 左端→順路→右端→逆路→左端にできた



- 端の様子は2通り
  - 出ていく先の方が入ってくる元より奥
  - 入ってくる元の方が出ていく先より奥

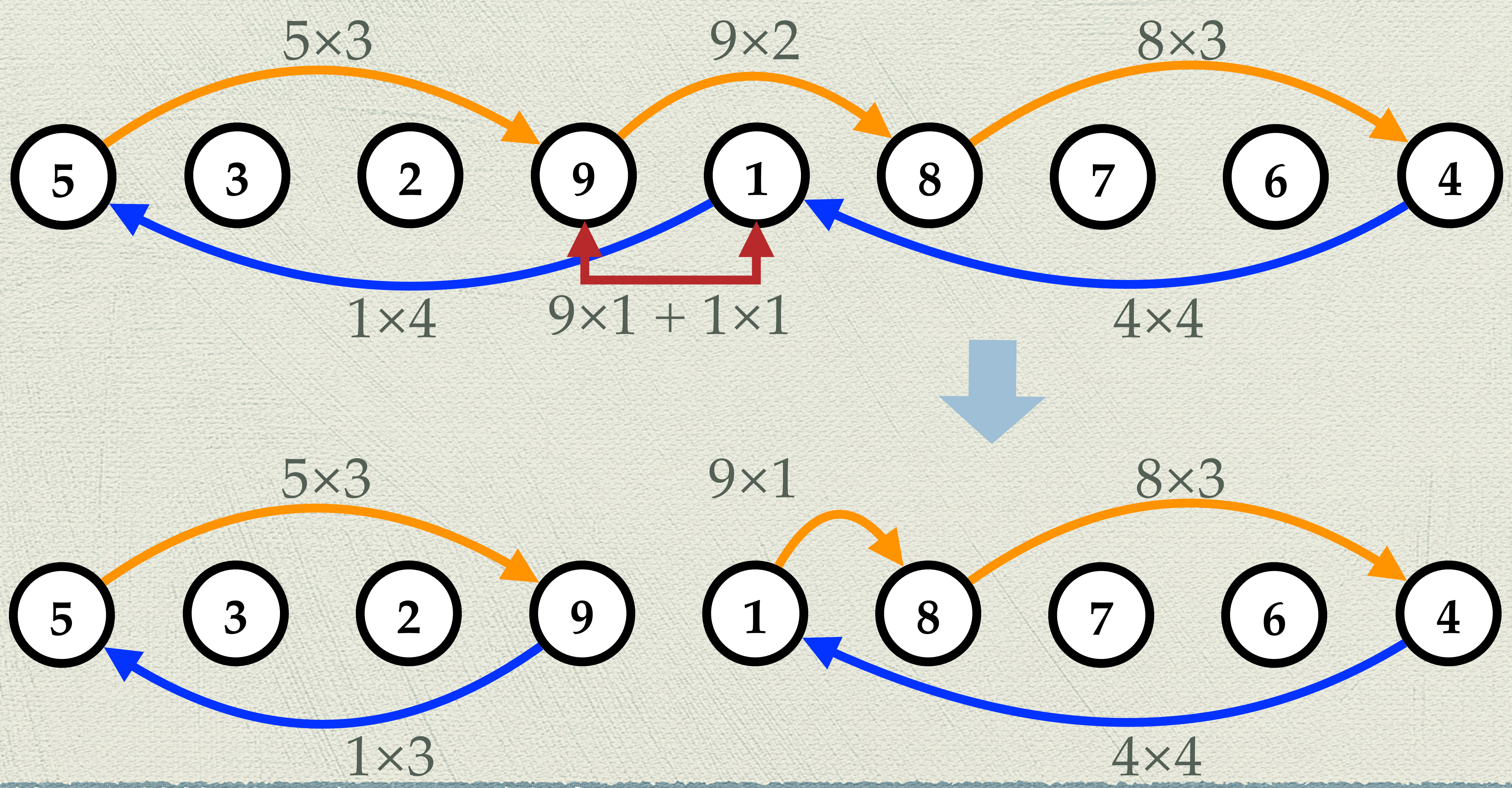
補題1: 入替最小=割当最小

- 出ていく先の方が入ってくる元より奥
  →入ってくる元と端を交換



補題1: 入替最小=割当最小

- 入ってくる元の方が出ていく先より奥
  →入ってくる元と出て行く先を交換



補題1: 入替最小=割当最小

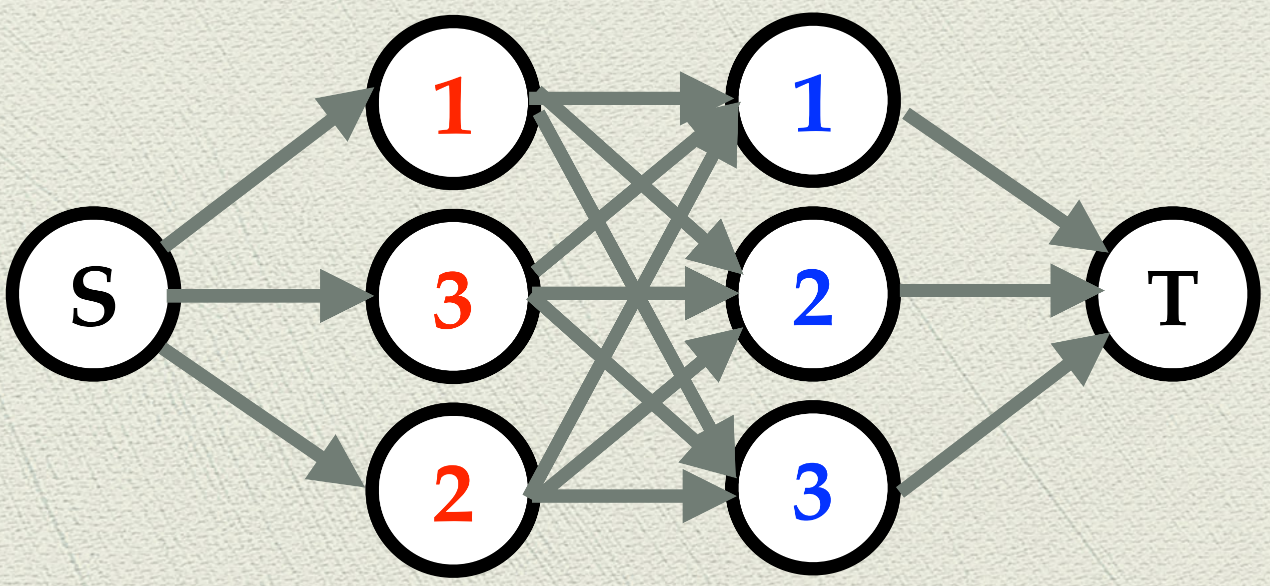
- 無駄なコストなしにサイクルを分解していく方法がある
- どの操作においてもサイクルの長さは真に小さくなるため、必ずすべて分解することができる

- 結論：<u>入替最小 = 割当最小</u>

補題2: 割当最小=最小費用流

- n個の要素とn個の位置との最小重み最大二部マッチングとなる
- p_iを位置jに割り当てるコストは$p_i \times |i-j|$
  - p_iからjにコスト$p_i \times |i-j|$の辺を張る
  - それぞれの要素・位置は1度しか使えない = 流量1

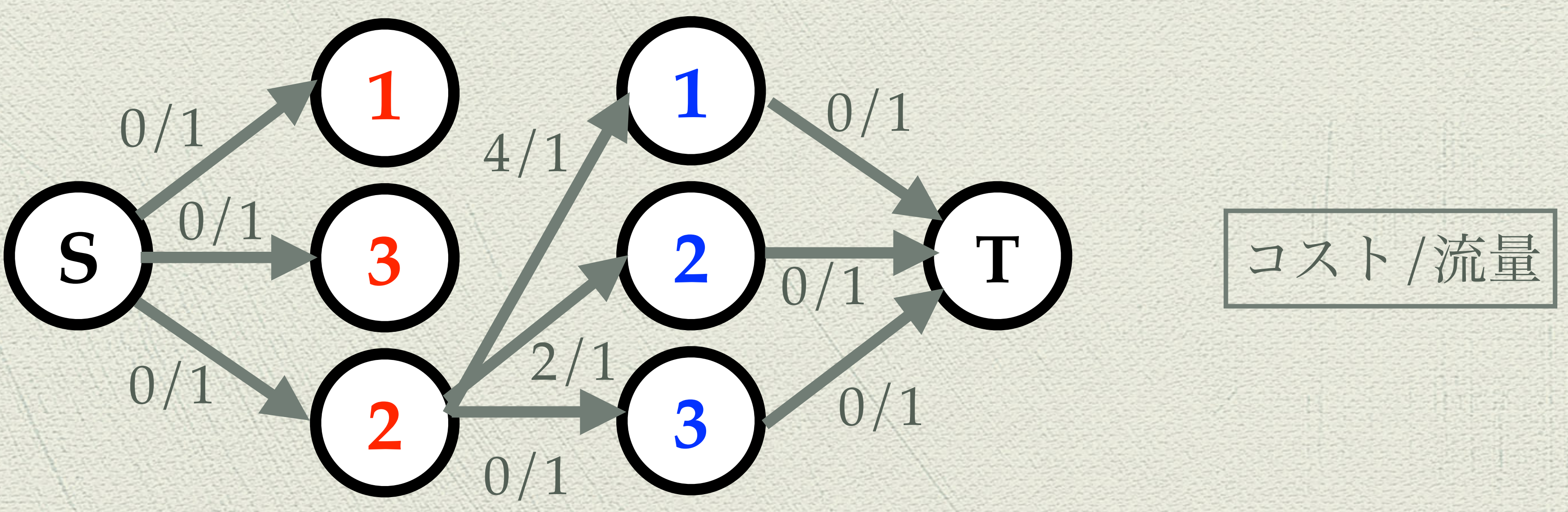
例: 132



補題2: 割当最小=最小費用流

- n個の要素とn個の位置との最小重み最大二部マッチングとなる
- p_iを位置jに割り当てるコストは$p_i \times |i-j|$
  - p_iからjにコスト$p_i \times |i-j|$の辺を張る
  - それぞれの要素・位置は1度しか使えない = 流量1

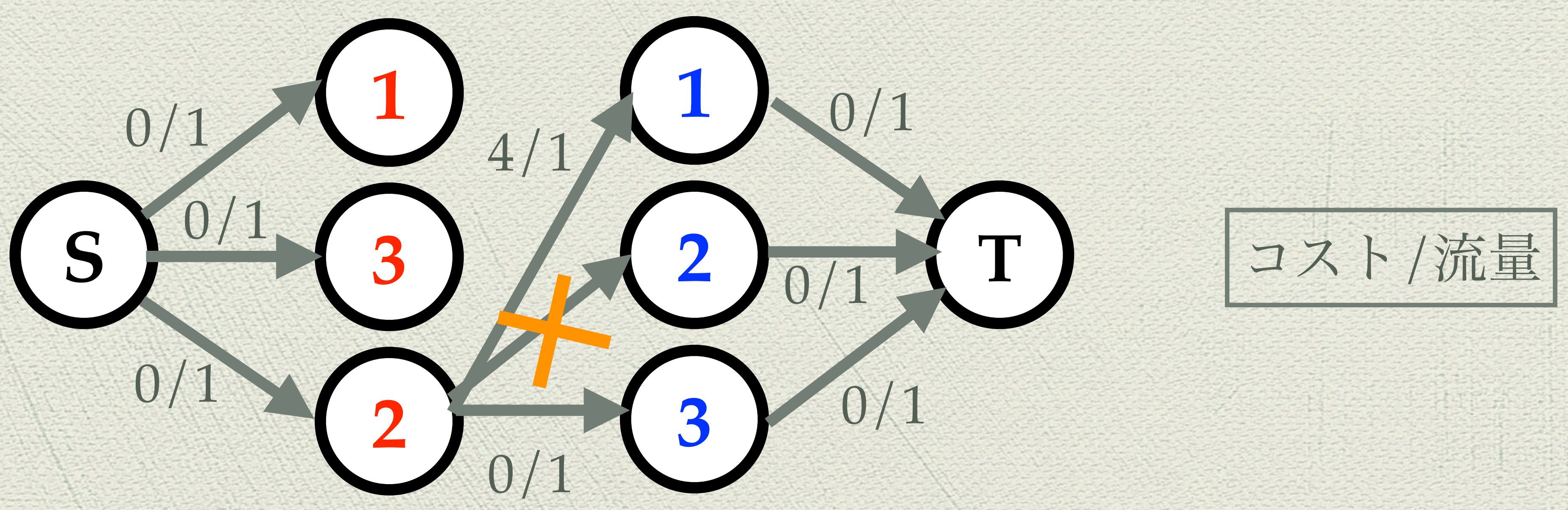
例: 132



補題2: 割当最小=最小費用流

- n個の要素とn個の位置との最小重み最大二部マッチングとなる
- p_iを位置jに割り当てるコストは$p_i \times |i-j|$
  - p_iからjにコスト$p_i \times |i-j|$の辺を張る
  - それぞれの要素・位置は1度しか使えない = 流量1

例: 132



writer解

- 井上(C++) 97行 $O(N^3)$
- 井上(C++) 90行 $O(N^3 \log N)$
- 辺が$N \times (N-1)$本あるグラフなので、primal-dual法でのダイクストラは$O(V^2)$の実装を用いた方が理論計算量上は高速になります

提出状況

- First Acceptance
  - on-site: lyozukky (02:13)
  - on-line: natsugiri (00:48)
  - on-rail: pepsin_amylase (00:39)
- 正答率 8/13 (61.5%)