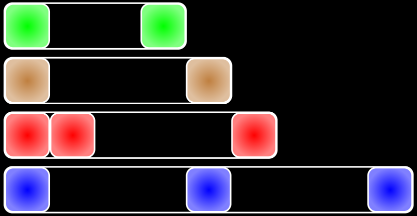
On Solving the Sparse Matrix Compression Problem Greedily

Dominik Köppl[1] Vincent Limouzy[2] Andrea Marino[3] Giulia Punzi[4]
Takeaki Uno[5]

[1]: University of Yamanashi
[2]: University Clermont Auvergne
[3]: University of Florence
[4]: University of Pisa
[5]: National Institute of Informatics

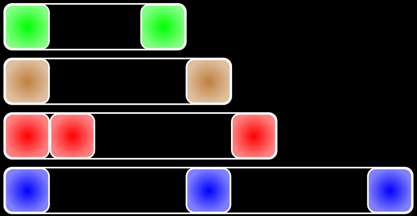


冬のLAシンポジウム2024 [24]

問題の設定

入力

- n 個の1次元ポリオミノ (=タイル)
- タイルには隙間があってもよい



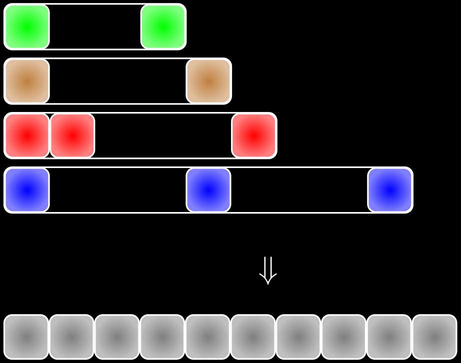
目標

- タイルを1次元結合タイルにする
- 隙間を埋めることができるが、埋めたブロックは重なってはならない
- 目的: 最短の結合タイルを構築

問題の設定

補題
隙間のない結合タイルは解である

証明.
ブロックは重ならないから □



決定問題

MINLENGTH すべてのタイルを長さ k の結合タイルに結合できるか?

MAXSHIFT すべてのタイルの最初のブロックが最初の列に配置された場合、最大シフトが k 以下の結合タイルを作成できるか?

MAXSHIFT はすでに Sparse Matrix Compression (SMC) problem として研究されていた

Garey+'79 は $k \geq 2$ の場合に SMC が $\mathcal{NP}$ 困難であることを示した
Bannai+'24 は幅が $\Omega(\lg n)$ の場合、両方の問題が $\mathcal{NP}$ 困難であることを示した

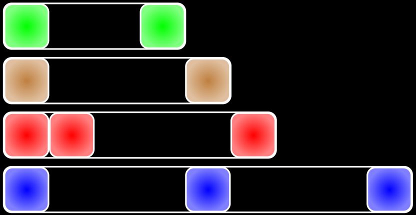
問題 (SMC, [Garey+'79, Chapter A4.2, Problem SR13])

入力：
- $n \times \ell$ 行列 $A[1..n][1..\ell]$、n 行 ℓ 列として、すべての $i \in [1..n]$, $j \in [1..\ell]$ に対して $A[i][j] \in \{0,1\}$ の要素を持つ
- 整数 $k \in [0..\ell \cdot (n-1)]$

目標: 以下の2つが存在できるかどうかを調べる:
- 整数配列 $C[1..\ell+k]$、$\forall i \in [1..\ell+k]$ について $C[i] \in [0..n]$
- シフト関数 $s : [1..n] \to [0..k]$ が存在し、$\forall i \in [1..n], \forall j \in [1..\ell]$ について $A[i][j] = 1 \Leftrightarrow C[s(i)+j] = i$ が成り立つ
- $A[0][j] = 0 \ \forall \ j$ であると仮定し、$C[i] = 0$ を設定することで、この要素が未割り当てであることをモデル化する

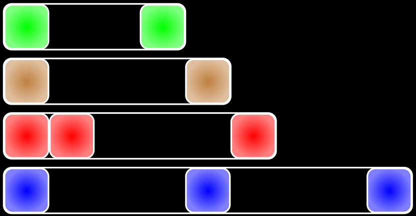
応用：
- 行列の圧縮 [Ziegler'77]
- 探索トライの実装 [Tarjan,Yao'79]
- コンパイラ [Aho+'86]
- ブルームフィルタ [Chang,Wu'91]

タイルから行列へ



$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

タイルから行列へ



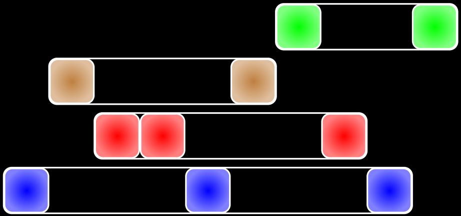
$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

解答：

- $s = [6, 1, 2, 0]$
- $C = [4, 2, 3, 3, 4, 2, 1, 3, 4, 1]$

$$B = \begin{pmatrix} & & & & & & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & & & & & \\ & & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & & & & \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & & & & & & \end{pmatrix}$$

タイルから行列へ



$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

解答：

- $s = [6, 1, 2, 0]$
- $C = [4, 2, 3, 3, 4, 2, 1, 3, 4, 1]$

$$B = \begin{pmatrix} & & & & & & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & & & & & \\ & & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & & & & \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & & & & & & \end{pmatrix}$$

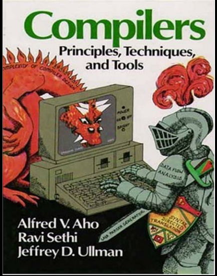
近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す

使用例: "Compilers: Principles, Techniques, and Tools" の節 3.9.8

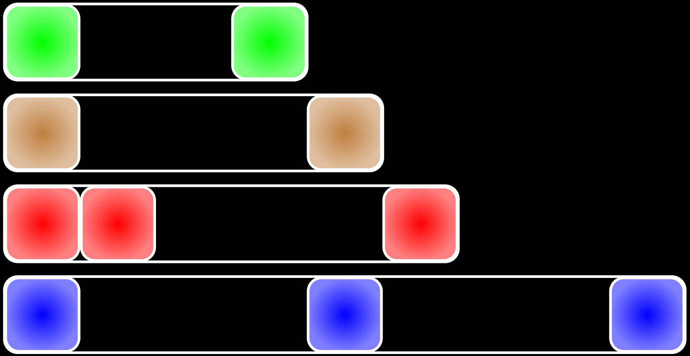
While we may not be able to choose *base* values so that no *next-check* entries remain unused, experience has shown that the simple strategy of assigning *base* values to states in turn, and assigning each *base*[*s*] value the lowest integer so that the special entries for state *s* are not previously occupied utilizes little more space than the minimum possible.



近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

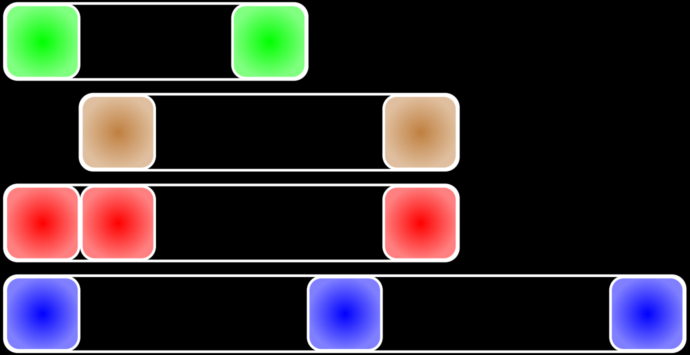
- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す



近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

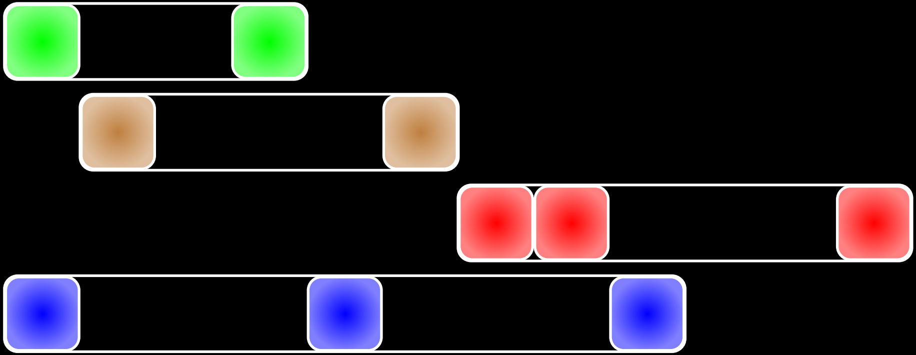
- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す



近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

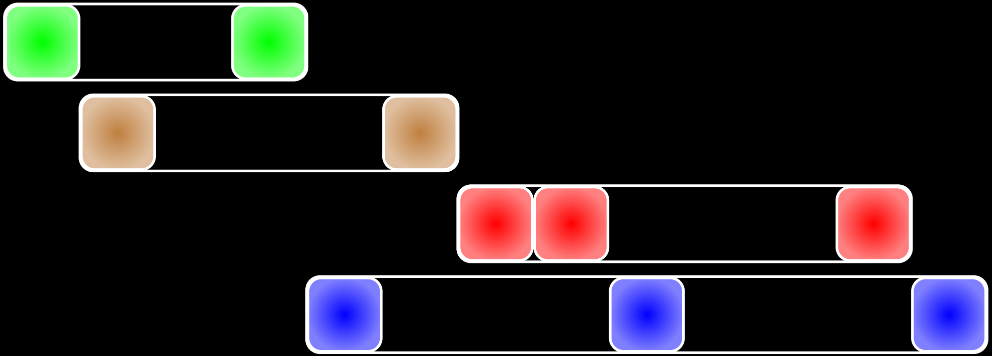
- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す



近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

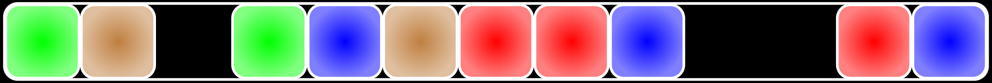
- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す



近似アルゴリズム

Ziegler'77: 貪欲アルゴリズム: 最初に最初のものを配置

- 最初のタイルを最初の位置に配置
- それ以降のタイルについて: 最も左に適合する位置に配置
- 繰り返す



- 近似比は本当に小さいか?
- 実際に、近似比は $\Theta(\sqrt{m})$, ここで m は最適値

下界: $\Omega(\sqrt{m})$ 近似比

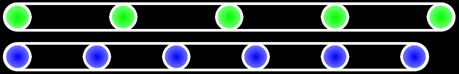
- 2種類のタイル: X と Y, $X = (1 \cdot 0^{k-2})^k$, $Y = (1 \cdot 0^{k-1})^k$
- $|X|, |Y| \in \Theta(k^2)$
- X 種類の個数: $k-2$, Y 種類の個数: $k-1$
- タイルの順序は $Y, X, Y, X, Y, X, Y, \ldots$
- 各配置は解に少なくとも $k^2 - k$ の長さを追加するので、合計の長さは $\Omega(k^3)$
- 逆に全ての X と Y は自分自身の中で結合して長さ $\Theta(k^2)$ の固まりにできる(最適値)
- 近似比は $\sqrt{m}$

貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける

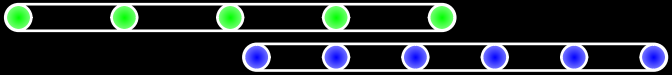
$Y =$

$X =$



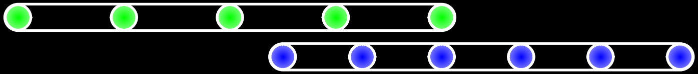
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける



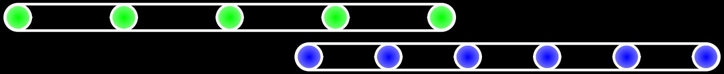
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける



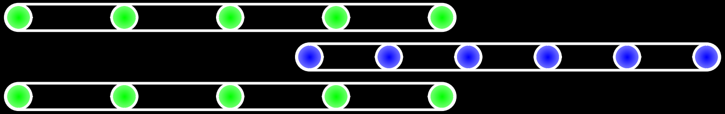
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合



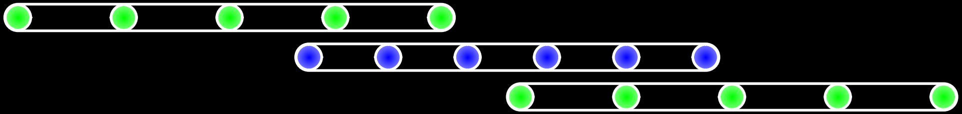
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合
- 次の Y が X と Y と衝突



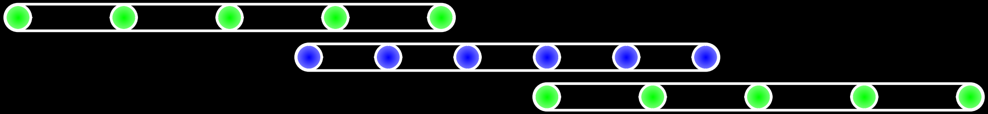
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合
- 次の Y が X と Y と衝突



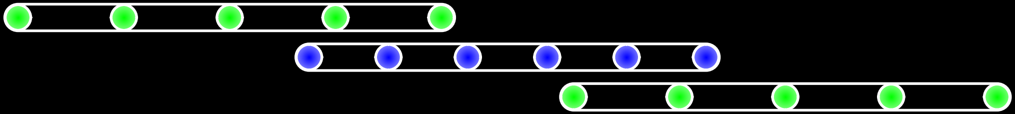
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合
- 次の Y が X と Y と衝突



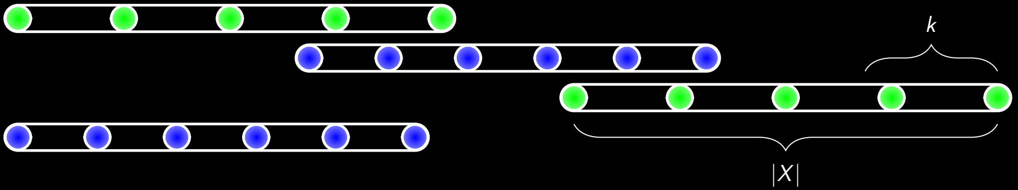
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合
- 次の Y が X と Y と衝突
- Y が X の最後の k 個の位置に適合する



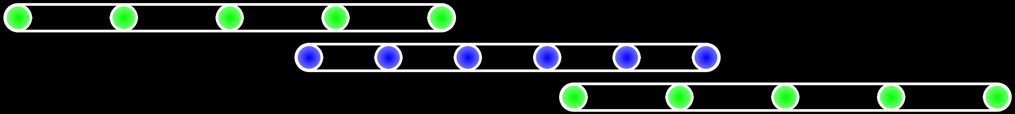
貪欲アルゴリズム

- Y から始めて X に最初に適合する場所を見つける
- X が Y の最後の k 個の位置の中に適合
- 次の Y が X と Y と衝突
- Y が X の最後の k 個の位置に適合する
- 再帰
- タイルを置くたびに文字列が $|X| - k$ だけ増える



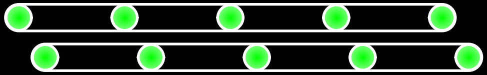
貪欲アルゴリズム: まとめ

- タイル種類 X と Y がそれぞれ k 個ある
- タイルの長さは $\Theta(k^2)$
- 各置いたタイルに対して、出力文字列の長さが $k^2 - k$ に増える
- 結局、出力の長さ: $\Omega(k^3)$
- 最短の出力は?



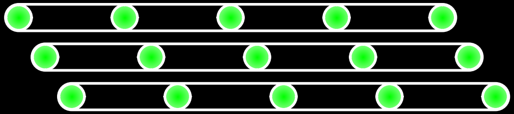
最適解

- まず全ての Y を揃える



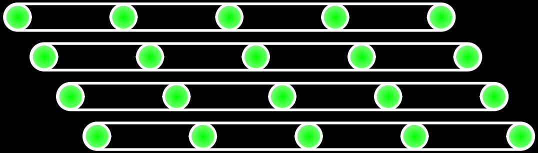
最適解

- まず全ての Y を揃える



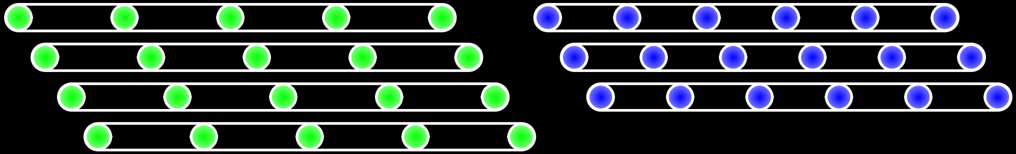
最適解

- まず全ての Y を揃える
- 全ての Y は完全に収まる



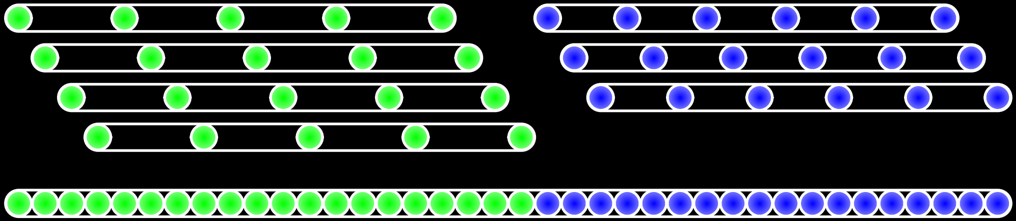
最適解

- まず全ての Y を揃える
- 全ての Y は完全に収まる
- 全ての X にも同じことが言える



最適解

- まず全ての Y を揃える
- 全ての Y は完全に収まる
- 全ての X にも同じことが言える
- 隙間がないよって、解は最適
- 解の長さは $(|X|+|Y|)+2k \in \Theta(k^2)$



recap

- 最適解の長さ $m \in \Theta(k^2)$
- 貪欲アルゴリズムの解の長さ: $\Omega(k^3)$
- 少なくとも $\Omega(k)$ 悪い、ここで $k \in \sqrt{m}$!

以下も示せる:

- 鳩の巣原理より、貪欲アルゴリズムは $\mathcal{O}(\sqrt{m})$ 以上悪くならない

$\Rightarrow$ 貪欲アルゴリズムの近似比は $\sqrt{m}$

- $n \times \ell$ 行列が与えられたとき、両問題を $\mathcal{O}(n^{2^\ell} \ell n 2^\ell n)$ 時間で正確に解ける

$\Rightarrow$ $\ell \in \mathcal{O}(\lg \lg n)$ のとき: 問題は $\mathcal{P}$ に属する

ご清聴ありがとうございました

未解決な問題

1. 順序を変わっても、$\Omega(\sqrt{m})$ の近似比下界が成り立つ？
2. より良い近似アルゴリズムの提案
3. FPT のアルゴリズムの存在
   - タイルの種類をパラメタにする
   - タイルのブロック ('1') の最大個数をパラマタにする
4. タイルの最大長さは
   - $\Omega(\lg n) \Rightarrow \mathcal{NP}$困難 Bannai+'24
   - $\mathcal{O}(\lg \lg n) \Rightarrow \mathcal{P}$
   - $\omega(\lg \lg n) \cap o(\lg n) \Rightarrow ?$