DNA計算に関するCoqモジュール開発

早川銀河
九州大学 大学院数理学府

溝口 佳寛
九州大学 マス・フォア・インダストリ研究所

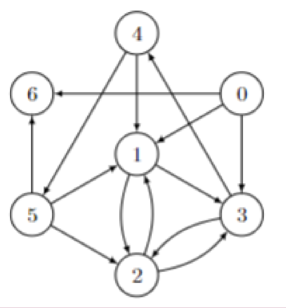
九州大学
KYUSHU UNIVERSITY

概要

DNA計算はDNAが持つ相補性を活かした計算理論であり,高い並列性による高速計算の実現やバイオテクノロジーと連携したシステムの実現などが期待されている.
DNAドミノと粘着演算,有限オートマトンをCoq/SSReflectのモジュールとして実装し,これを用いて有限オートマトンからスティッカーシステムへの変換関数を定義してその生成言語が元の有限オートマトンの受理言語と等しいことを形式的に証明した.

DNA Computing

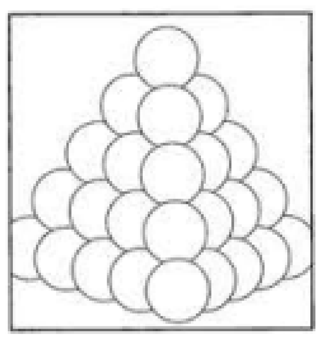
L.AdlemanはDNA分子の相補的な二本鎖構造に着目し,塩基配列でグラフのノードとエッジを表現することにより7つの頂点と14の有向辺からなるハミルトン路問題 (Hamiltonian Path Problem; HPP)を解くことに成功した[6]. HPPはNP完全問題の一つで,DNA計算の可能性を切り開く重要な出来事となった.



Proof Assistant

数学の証明を形式的に記述し検証するソフトウェア
論理的な推論規則に基づく厳密な証明が可能となる他,以下のような手計算では困難なほど複雑な定理の証明にも用いられる.

- ・四色定理 (Coq/SSReflect, 2005)[4]
- ・ケプラー予想 (Isabelle/HOL, 2014)[5]



Automaton

有限オートマトン:$M = (Q, \Sigma, \delta, s_0, F)$
Q: 状態の有限集合,Σ:アルファベット,$\delta: (Q \times \Sigma) \to Q$ 遷移関数,$q_0 \in Q$: 初期状態,$F \subseteq Q$:受理状態

```
(*有限オートマトン*)
Structure automaton {state symbol:finType} :=
  {delta: state -> symbol -> state; init: state; final: {set state}}.

(*遷移関数の反射的推移閉包*)
Fixpoint dstar {state symbol:finType}
  (delta:state->symbol->state)(q:state)(str:seq symbol):state :=
  match str with nil => q|h::str' => dstar delta (delta q h) str’ end.

(*受理判定*)
Definition accept {state symbol:finType}(M:@automaton state symbol)
  (str:seq symbol):bool := dstar (delta M) (init M) str¥in final M.
```

DNA Domino

DNAドミノ $\mathfrak{D}_\rho(V) = S_\rho(V) \cup \mathrm{WK}_\rho(V) \cup L_\rho(V) \cup R_\rho(V) \cup LR_\rho(V)$

DNA分子の二本鎖構造を抽象化したもので,アルファベットVとその相補的対応関係$\rho \subseteq V \times V$が与えられたとき、二本鎖部と粘着末端の組み合わせで定義される.

$S_\rho(V) = \binom{V^*}{\lambda} \cup \binom{\lambda}{V^*}$ （粘着末端）

$WK_\rho(V) = \begin{bmatrix} V^+ \\ V^+ \end{bmatrix}_\rho$ （二本鎖部）

$L_\rho(V) = \left(\binom{V^+}{\lambda} \cup \binom{\lambda}{V^+}\right)\begin{bmatrix} V^+ \\ V^+ \end{bmatrix}_\rho$

$R_\rho(V) = \begin{bmatrix} V^+ \\ V^+ \end{bmatrix}_\rho \left(\binom{V^+}{\lambda} \cup \binom{\lambda}{V^+}\right)$

$LR_\rho(V) = \left(\binom{V^+}{\lambda} \cup \binom{\lambda}{V^+}\right)\begin{bmatrix} V^+ \\ V^+ \end{bmatrix}_\rho \left(\binom{V^+}{\lambda} \cup \binom{\lambda}{V^+}\right)$

```
(*二本鎖部*)
Structure wk{symbol:finType}{rho:seq (symbol*symbol)} :=
  {str:seq(symbol*symbol);nilP:str<>nil;rhoP:all(fun p=>p¥in rho)str}.

(*粘着末端*)
Structure stickyend{symbol:finType}:=
  {is_upper : bool; end_str : seq symbol; end_nilP : end_str <> nil}.

(*ドミノ*)
Inductive domino{symbol:finType}{rho:Rho symbol}:=
  |null
  |Simplex(s:@stickyend symbol)
  |WK(w:@wk symbol rho)
  |L(l:@stickyend symbol)(w:@wk symbol rho)
  |R(r:@stickyend symbol)(w:@wk symbol rho)
  |LR(l r:@stickyend symbol)(w:@wk symbol rho).
```

Sticker System

粘着演算 $\mu: \mathfrak{D} \times \mathfrak{D} \to \mathfrak{D} \cup \{\perp\}$

DNA分子は水素結合と酵素の働きにより,相補的な粘着末端を持つ分子同士で結合する.
DNAドミノの粘着演算μを現実のDNA分子に沿う形で定義する.

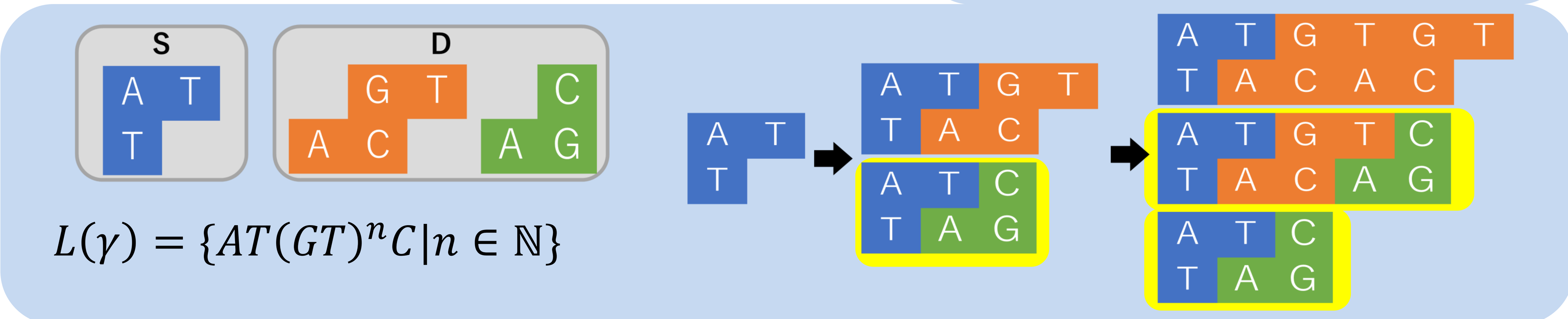
スティッカーシステム $\gamma = (V, \rho, S, D)$

V: 記号の有限集合, $\rho \subseteq V \times V, S \subset LR_\rho(V)$(有限集合), $D \subset \mathfrak{D}_\rho(V) \times \mathfrak{D}_\rho(V)$(有限集合)
スティッカーシステムはドミノの粘着演算を活かした言語生成器であり，その生成言語$L(\gamma)$は以下のように定義される.

$$L(\gamma) = \left\{ w \in V^* \,\middle|\, \begin{bmatrix} w \\ w' \end{bmatrix}_\rho \in \mu_D^n(S), n \in \mathbb{N} \right\}$$

$$\mu_D(X) \coloneqq \{\mu(l, \mu(x, r)) \,|\, x \in X, (l, r) \in D\}$$

例）
$V = \{A, T, G, C\}$
$\rho = \{(A, T), (T, A), (G, C), (C, G)\}$
$S = \left\{\begin{bmatrix} A \\ T \end{bmatrix}\binom{T}{\lambda}\right\}$
$D = \left\{\left(\binom{\lambda}{\lambda}, \binom{\lambda}{A}\begin{bmatrix} G \\ C \end{bmatrix}\binom{T}{\lambda}\right), \left(\binom{\lambda}{\lambda}, \binom{\lambda}{A}\begin{bmatrix} C \\ G \end{bmatrix}\right)\right\}$
$\gamma = (V, \rho, S, D)$




```
(*粘着演算*)
Definition mu{symbol:finType}{rho:seq(symbol*symbol)}
  (x y:@domino symbol rho):=
  match x,y with
    |null,_ => Some y
    |_,null => Some x
    |Simplex s1,WK w2 => Some (L s1 w2)
(*中略*)
    |LR l1 w1 (Se false r1 _),LR (Se true l2 _) w2 r2 =>
      if size r1 == size l2 then
        match mu_end rho l2 r1 with
          |Some w => Some (LR l1 w1#w#w2 r2)
          |None => None
        end
      else
        None
    |_,_ => None
  end.

(*スティッカーシステム*)
Structure sticker{symbol:finType}{rho:seq (symbol*symbol)} := {
  start : seq (@domino symbol rho);
  extend : seq (@domino symbol rho*@domino symbol rho)}.
```

スティッカーシステムの計算能力

任意のオートマトン$M = (Q, \Sigma, \delta, q_0, F)$が与えられたとき、以下の手順に従って右側スティッカーシステムγ_Mを構成することで$L(M) \subseteq L(\gamma_M)$を示せる[1].

$$\gamma_M = (Q, \rho, S, D)$$

$$\rho = \{(a, a) | a \in \Sigma\},\ S = S_1 \cup S_2,\ D = D_1 \cup D_2$$

$$S_1 = \left\{ \begin{bmatrix} x \\ x \end{bmatrix}_\rho \,\middle|\, x \in L(M), |x| < |Q| \right\},\ S_2 = \left\{ \begin{bmatrix} x \\ x \end{bmatrix}_\rho \binom{u}{\lambda} \,\middle|\, x, u \in \Sigma^*, |xu| = |Q| + 1, |u| = \delta^*(q_0, xu) \right\}$$

$$D_1 = \left\{ \binom{\lambda}{\lambda}, \binom{\lambda}{v} \begin{bmatrix} x \\ x \end{bmatrix}_\rho \binom{u}{\lambda} \,\middle|\, x, u, v \in \Sigma^*, |xu| = |Q| + 1, |u| = \delta^*(|v|, xu) \right\}$$

$$D_2 = \left\{ \binom{\lambda}{\lambda}, \binom{\lambda}{v} \begin{bmatrix} x \\ x \end{bmatrix}_\rho \,\middle|\, x, v \in \Sigma^*, 0 < |x| \le |Q|, 0 < |v| \le |Q|, \delta^*(|v|, x) \in F \right\}$$

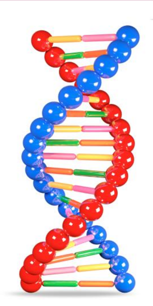
この変換関数(Aut_to_Stk: automaton → sticker)をCoqで定義し,
$^{\forall}M \in automaton, L(M) \subseteq L(Aut\_to\_Stk(M))$ を形式的に証明した.

```
Theorem REG_RSL{state symbol : finType}(M:@automaton state symbol)
  (s:seq symbol) : s <> nil -> exists n:nat, forall m : nat, n <= m ->
  accept M s = ( s ¥in (ss_language_prime m (Aut_to_Stk M))).
Proof.
(*中略*)
Qed.
```

関数定義や関連補題を含めた形式証明全文はおよそ2800行にわたる.
関連ファイルおよび説明資料を以下のページにて公開している.
https://github.com/KyushuUniversityMathematics/CoqSticker

今後の課題

DNA計算のモデルは他にも**両側スティッカーシステム**や**スプライシングシステム**など多数提案されており，中にはチューリングマシンと同等の計算能力を持つものも存在する.
今後，それらの計算モデルも実装できるようモジュールを拡張し，DNA計算に関する諸定理に形式的な証明を与えたい.



参考文献

[1] H. Tanaka, et.al, Formal Proofs for Automata and Sticker Systems, CANDAR’13, 2013.12.
[2] 萩原学,アフェルト･レナルド, Coq/ SSreflect/ MathCompによる形式証明, 森北出版, 2018.
[3] G.パウン,他,DNAコンピューティング,シュプリンガー, 1999.
[4] G. Gonthier, Formal proof-the four-color theorem, Notices of the AMS, 2008.
[5] T. Hales, et.al, A Formal Proof of The Conjecture, Forum of Mathematics, Pi , 2017.
[6] L. M. Adleman, Computing with DNA, Scientific Amelican,1998.
[7] The Coq Proof Assistant, https://coq.inria.fr/ .

※PPL2025 (第27回プログラミングおよびプログラミング言語ワークショップ) 2025.3.5-7