Site-Directed Insertion and Deletion

Da-Jung Cho

CNRS & LSV, ENS Paris-Saclay
LRI, Université Paris-Sud

MeFoSyLoMa Seminar
Outline

1. Introduction
   - Background
   - Definition
   - Problems

2. Main Results
   - Closure Properties
   - Decidability

3. Conclusion
   - Summary
   - Future Works
Outline

1 Introduction
   • Background
   • Definition
   • Problems

2 Main Results
   • Closure Properties
   • Decidability

3 Conclusion
   • Summary
   • Future Works
Site-Directed Insertion and Deletion from Molecular Biology

- The site-directed insertion (deletion) operation models *site-directed mutagenesis*.
- It creates an insertion (a deletion) on specific sites of a given gene sequence under enzymatic activities.
(a) Site-directed insertion on plasmid

(b) Site-directed deletion on plasmid
Outline

1. Introduction
   - Background
   - Definition
   - Problems

2. Main Results
   - Closure Properties
   - Decidability

3. Conclusion
   - Summary
   - Future Works
Site-Directed Insertion

Definition

Given two strings $x = x_1uvx_2$ and $y = uwv$, the site-directed insertion of $y$ into $x$ is defined to be

$$x \xleftarrow{\text{sdi}} y = \{x_1uwvx_2 \mid x_1, x_2, w \in \Sigma^* \text{ and } u, v \neq \lambda\}.$$  

- for a string $y = uwv$, we say $(u, v)$ is an outfix of $y$
- an outfix $(u, v)$ of $y$ is an insertion guide of $x$ if $x \xleftarrow{\text{sdi}} y \neq \emptyset$.
- extend site-directed insertion to languages

$$L_1 \xleftarrow{\text{sdd}} L_2 = \bigcup_{w_i \in L_i, i = 1, 2} w_1 \xleftarrow{\text{sdi}} w_2.$$
Site-Directed Insertion

Definition

Given two strings $x = x_1uvx_2$ and $y = uwv$, the site-directed insertion of $y$ into $x$ is defined to be

$$x \xleftarrow{\text{sdi}} y = \{x_1uwvx_2 \mid x_1, x_2, w \in \Sigma^* \text{ and } u, v \neq \lambda\}.$$
Site-Directed Deletion

Definition

Given two strings \( x = x_1 u w v x_2 \) and \( y = uv \), the site-directed deletion from a string \( x \) by \( y \) is

\[
x \xrightarrow{sdd} y = \{ x_1 uv x_2 \mid x_1, x_2, w \in \Sigma^* \text{ and } u, v \neq \lambda \}.
\]

- A string \( y \) is a deletion guide of \( x \) if \( x \xrightarrow{sdd} y \neq \emptyset \)
- \( L_1 \xrightarrow{sdd} L_2 = \bigcup_{w_i \in L_i, i=1,2} w_1 \xrightarrow{sdd} w_2 \)
Site-Directed Deletion

Definition

Given two strings $x = x_1 u w v x_2$ and $y = u v$, the site-directed deletion from a string $x$ by $y$ is

$$x \xleftarrow{sdd} y = \{x_1 u v x_2 \mid x_1, x_2, w \in \Sigma^* \text{ and } u, v \neq \lambda\}.$$

(a) Ordinary deletion on $x$

(b) Site-directed deletion on $x$ and $y$
Iterated Site-Directed Insertion/Deletion

Definition

Site-directed insertion of $L$ is inductively defined as

$$SDI^{(0)}(L) = L, \text{ and } SDI^{(i+1)}(L) = SDI^{(i)}(L) \xleftarrow{sdi} SDI^{(i)}(L).$$

The iterated site-directed deletion $SDI^*$ of $L$ is

$$SDI^*(L) = \bigcup_{i=1}^{\infty} SDI^{(i)}(L).$$
Iterated Site-Directed Insertion/Deletion

Definition

Site-directed deletion of $L$ is inductively defined as

$$SDD^{(0)}(L) = L,$$
and
$$SDD^{(i+1)}(L) = SDD^{(i)}(L) \xleftarrow{sdd} SDD^{(i)}(L).$$

The iterated site-directed deletion $SDD^*$ of $L$ is

$$SDD^*(L) = \bigcup_{i=1}^{\infty} SDD^{(i)}(L).$$
Outline

1. Introduction
   - Background
   - Definition
   - Problems

2. Main Results
   - Closure Properties
   - Decidability

3. Conclusion
   - Summary
   - Future Works
In Formal Language Theory

Researchers in formal language theory characterize the biological phenomena into OPERATIONS on strings.

- Searls, "The computational linguistics of biological sequences", *Artificial Intelligence and Molecular Biology*, 1993
- Dassow et al., "Context-free evolutionary grammars and the structural language of nucleic acids", *Biosystems*, 1997
- Dassow et al., "Operations and language generating devices suggested by the genome evolution", *Theoretical Computer Science*, 2002
- Leupold et al., "Formal languages arising from gene repeated duplication", *Theoretical Computer Science*, 2004
- Enaganti et al., "A formal language model of DNA polymerase enzymatic activity", *Fundamenta Informaticae*, 2015
In Formal Language Theory

A formal language is
- a set of strings over $\Sigma$, a finite alphabet,
- generated by grammar,
- classified in the Chomsky hierarchy
Theoretical Problems

- **Closure**: Decide whether or not languages in the Chomsky hierarchy are closed under the operation ♣

  Given a regular language $L$, is ♣($L$) REGULAR?

- **Membership problem**: Decide whether or not a given string $x$ belongs to the language ♣($L$)

- **♣-closed decidability**: Decide whether or not a given language $L$ produces no strings outside of $L$
Theoretical Problems

- **Closure**: Decide whether or not languages in the Chomsky hierarchy are closed under the operation ♣

- **Membership problem**: Decide whether or not a given string $x$ belongs to the language $♣(L)$

  Given $x$ and $L$, is $x \in ♣(L)$?

- **♣-closed decidability**: Decide whether or not a given language $L$ produces no strings outside of $L$
Theoretical Problems

- **Closure**: Decide whether or not languages in the Chomsky hierarchy are closed under the operation ♣

- **Membership problem**: Decide whether or not a given string $x$ belongs to the language ♣($L$)

- **♣-closed decidability**: Decide whether or not a given language $L$ produces no strings outside of $L$

Given $L$, is $L$ ♣-closed, in other words, ♣($L$) ⊆ $L$?
Outline

1 Introduction
   - Background
   - Definition
   - Problems

2 Main Results
   - Closure Properties
   - Decidability

3 Conclusion
   - Summary
   - Future Works
Closure Properties

**Theorem**

If $L_1$ and $L_2$ are regular, then $L_1 \xleftarrow{sdd} L_2$ is regular.

Let $L_1$ and $L_2$ be recognized by NFAs $A = (Q_A, \Sigma, \delta, q_0, F_A)$ and $B = (Q_B, \Sigma, \gamma, p_0, F_B)$, respectively.

We construct an NFA $C = (Q_C, \Sigma, \omega, s, F_C)$ recognizing all strings in $L(A) \xleftarrow{sdd} L(B)$.

Suppose that $x_1uwvx_2 \in L(A)$, $uv \in L(B)$ and $x_1uvx_2 \in L(A) \xleftarrow{sdd} L(B)$.

$$Q_C = Q_A \times (\{\spadesuit, \heartsuit\} \cup Q_B) \cup \tilde{Q}_A \times Q_B$$

\[
\begin{align*}
(Q_A, \spadesuit) & \quad (Q_A, Q_B) & \quad (\tilde{Q}_A, Q_B) & \quad (Q_A, \heartsuit) \\
x_1 & \quad u & \quad v & \quad x_2
\end{align*}
\]
If $L_1$ and $L_2$ are regular, then $L_1^{sdd} \leftarrow L_2$ is regular.

$x_1uvx_2 \in x_1uwvx_2^{sdd}uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$
Closure Properties of Site-Directed Deletion

**Theorem**

*If $L_1$ and $L_2$ are regular, then $L_1 \overset{sdd}{\leftarrow} L_2$ is regular.*

$x_1uvx_2 \in x_1uwvx_2 \overset{sdd}{\leftarrow} uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$
Closure Properties of Site-Directed Deletion

Theorem

If $L_1$ and $L_2$ are regular, then $L_1 \overset{sdd}{\leftarrow} L_2$ is regular.

$x_1uvx_2 \in x_1uwvx_2 \overset{sdd}{\leftarrow} uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$

NFA $A$

NFA $B$

NFA $C$
Closure Properties of Site-Directed Deletion

Theorem

If \( L_1 \) and \( L_2 \) are regular, then \( L_1^{sdd} \leftarrow L_2 \) is regular.

\[
x_1 uv x_2 \in x_1 uwvx_2 \overset{sdd}{\leftarrow} uv, \text{ where } x_1 uwvx_2 \in L(A) \text{ and } uv \in L(B)
\]
**Closure Properties of Site-Directed Deletion**

**Theorem**

If $L_1$ and $L_2$ are regular, then $L_1 \xleftarrow{sdd} L_2$ is regular.

$x_1uvx_2 \in x_1uwvx_2 \xleftarrow{sdd} uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$
Theorem

If $L_1$ and $L_2$ are regular, then $L_1^{sdd} \leftarrow L_2$ is regular.

$x_1uvx_2 \in x_1uwvx_2^{sdd} \leftarrow uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$

NFA $A$

NFA $B$

NFA $C$
Closure Properties of Site-Directed Deletion

**Theorem**

If $L_1$ and $L_2$ are regular, then $L_1 \xleftarrow{sdd} L_2$ is regular.

$x_1uvx_2 \in x_1uwvx_2 \xleftarrow{sdd} uv$, where $x_1uwvx_2 \in L(A)$ and $uv \in L(B)$
There exists context-free languages $L_1$ and $L_2$ such that $L_1 \xleftarrow{sdd} L_2$ is not context-free.

$L_1 = \{ a^n b^n c^m \# \mid m, n \geq 1 \}$,
$L_2 = \{ b^n c^n \# \mid n \geq 1 \}$.

Then, we have

$$(L_1 \xleftarrow{sdd} L_2) \cap (a^* b^* c^* \#) = \{ a^n b^n c^n \# \mid n \geq 1 \}$$
Closure Properties

About site-directed insertion:

- For regular languages $L_1$ and $L_2$, $L_1 \overset{sdi}{\leftarrow} L_2$ is regular.
- For context-free languages $L_1$ and $L_2$, $L_1 \overset{sdi}{\leftarrow} L_2$ is not context-free.
- For context-free language $L_1$ and regular language $L_2$, $L_1 \overset{sdi}{\leftarrow} L_2$ and $L_2 \overset{sdi}{\leftarrow} L_1$ are context-free.
- There exists a finite language $L$ such that $\text{SDI}^*(L)$ is non-regular.
- **Open**: Find a regular (or a finite) language $L$ such that $\text{SDI}^*(L)$ is not context-free.
Closure Properties

About site-directed deletion:

- For regular languages $L_1$ and $L_2$, $L_1 \leftarrow L_2$ is regular.
- For context-free languages $L_1$ and $L_2$, $L_1 \leftarrow L_2$ is not context-free.
- For context-free language $L_1$ and regular language $L_2$, $L_1 \leftarrow L_2$ and $L_2 \leftarrow L_1$ are context-free.
- There exists a context-free language $L$ such that $\text{SDD}^*(L)$ is not context-free.
- Open: Find a regular language $L$ such that $\text{SDD}^*(L)$ is not regular.
Outline

1. Introduction
   - Background
   - Definition
   - Problems

2. Main Results
   - Closure Properties
   - Decidability

3. Conclusion
   - Summary
   - Future Works
Decidability

**Theorem**

*Given three strings* $x, y, z \in \Sigma^*$, *we can determine whether or not* $z \in x \xleftarrow{sdd} y$ *in* $O(n)$ *time, where* $|x| = n$ *and* $|x| \geq |z| \geq |y| \geq 2$.

Suppose that there exist $x, y, \text{and} z$ such that $z \in x \xleftarrow{sdd} y$. 

![Diagram of site-directed insertion and deletion](image)

---

*Cho, Da-Jung*  
Site-Directed Insertion and Deletion  
November 9, 2018  21 / 35
Decidability (continued)

**Theorem**

*Given three strings* \( x, y, z \in \Sigma^* \), *we can determine whether or not* \( z \in x \xleftarrow{sdd} y \) *in* \( \mathcal{O}(n) \) *time, where* \(|x| = n\) *and* \(|x| \geq |z| \geq |y| \geq 2\).*

Scan both ends of \( x \) and \( z \) until a mismatch occurs.
Decidability (continued)

**Theorem**

*Given three strings* $x, y, z \in \Sigma^*$, *we can determine whether or not* $z \in x \overset{sdd}{\leftarrow} y$ *in* $O(n)$ *time, where* $|x| = n$ *and* $|x| \geq |z| \geq |y| \geq 2$.

Scan both ends of $x$ and $z$ until a mismatch occurs.

```
x
```

Find the longest matching prefix

```
z
```

Find the longest matching suffix
Decidability (continued)

**Theorem**

Given three strings $x, y, z \in \Sigma^*$, we can determine whether or not $z \in x^{sdd} y$ in $\mathcal{O}(n)$ time, where $|x| = n$ and $|x| \geq |z| \geq |y| \geq 2$.

$$x[1 : i] = z[1 : i] \text{ and } x[n-j+1 : n] = z[l-j+1 : l]$$
Decidability (continued)

**Theorem**

*Given three strings* $x, y, z \in \Sigma^*$, *we can determine whether or not* $z \in x \overset{sdd}{\leftarrow} y$ *in* $O(n)$ *time, where* $|x| = n$ *and* $|x| \geq |z| \geq |y| \geq 2$.

We check whether or not $y = uv$ is a substring of $z$. 

![Diagram showing the theorem with strings $y$, $z$, and $x_1 x_2$]

Cho, Da-Jung  
Site-Directed Insertion and Deletion  
November 9, 2018  
22 / 35
Theorem

Given three strings \( x, y, z \in \Sigma^* \), we can determine whether or not \( z \in x \xleftarrow{sdd} y \) in \( \mathcal{O}(n) \) time, where \( |x| = n \) and \( |x| \geq |z| \geq |y| \geq 2 \).

A prefix of \( y \) should be a suffix of the longest matching prefix of \( z \),
A suffix of \( y \) should be a prefix of the longest matching suffix of \( z \).
Decidability (continued)

Theorem

Given three strings \( x, y, z \in \Sigma^* \), we can determine whether or not \( z \in x \overset{sdd}{\leftarrow} y \) in \( \mathcal{O}(n) \) time, where \( |x| = n \) and \( |x| \geq |z| \geq |y| \geq 2 \).

We check for an occurrence of \( y \) within \( z[l-(j+m)+2 : i+m-1] \).
Decidability (continued)

**Theorem**

*Given three strings* $x, y, z \in \Sigma^*$, *we can determine whether or not* $z \in x \overset{sdd}{\leftarrow} y$ *in* $O(n)$ *time, where* $|x| = n$ *and* $|x| \geq |z| \geq |y| \geq 2$.

KMP pattern matching algorithm returns 1 if $y$ occurs in the search-range.

$y$ occurs in the search-range of $z$

$l - (j+m) + 2 \quad \quad i + m - 1$
Decidability (continued)

**Theorem**

*Given three strings* $x, y, z \in \Sigma^*$, *we can determine whether or not* $z \in x \xrightarrow{sdd} y$ *in* $\mathcal{O}(n)$ *time, where* $|x| = n$ *and* $|x| \geq |z| \geq |y| \geq 2$.
Decidability

Theorem

Given two strings $x$ and $y$, we can determine whether or not $x \xrightarrow{sdd} y \neq \emptyset$ in $O(n)$ time, where $|x| = n$, $|y| = m$ and $m \leq n$.

Determine whether or not there exist two substrings $u$ and $v$ of $x$, whose catenation is $y$. 

![Diagram showing the decidability of site-directed insertion and deletion]
Decidability

Theorem

Given two strings \( x \) and \( y \), we can determine whether or not \( x^{\text{sdd}} \xleftarrow{} y \neq \emptyset \) in \( \mathcal{O}(n) \) time, where \( |x| = n, |y| = m \) and \( m \leq n \).

Construct two Aho-Corasick automata \( A_P \) (\( A_S \)) with all prefixes of \( y(y^R) \), and run them on \( x \).

\[
\begin{align*}
\text{Pattern: prefixes of } y & \quad \text{Pattern: prefixes of } y^R \\
\text{Result: pairs } (j, g) & \quad \text{Result: pairs } (k, h)
\end{align*}
\]

\[
x[j - g + 1 : j] = y[1 : g] \quad \text{and} \quad x[k : k + h - 1] = y[m - h + 1 : m]
\]
Decidability

Theorem

Given two strings $x$ and $y$, we can determine whether or not $x \overset{sdd}{\leftarrow} y \neq \emptyset$ in $O(n)$ time, where $|x| = n$, $|y| = m$ and $m \leq n$.

Construct two Aho-Corasick automata $A_P$ ($A_S$) with all prefixes of $y(y^R)$, and run them on $x$. 

![Diagram of Aho-Corasick automata with strings x, y, and their prefixes u, v, i, j, k, g, h.](image)
Decidability

Theorem

Given two strings $x$ and $y$, we can determine whether or not $x \xleftarrow{sdd} y \neq \emptyset$ in $O(n)$ time, where $|x| = n$, $|y| = m$ and $m \leq n$.

Construct two Aho-Corasick automata $A_P (A_S)$ with all prefixes of $y(y^R)$, and run them on $x$.

1. $y[1 : g] = x[j - g + 1 : j]$
2. $y[m - h + 1 : m] = x[k : k + h - 1]$
3. $j - g + i \leq k + h - m + i$
4. $i \leq g$
5. $m - i \leq h$
Decidability

Definition

A language $L$ is
- closed (or, \textit{sdi-closed}) under site-directed insertion if $L^\text{sdi} \subseteq L$,
- closed (or, \textit{sdd-closed}) under site-directed deletion if $L^\text{sdd} \subseteq L$.

For a given regular language $L$,
- it is \textit{decidable} to determine whether or not $L$ is \textit{sdi-closed}
- it is \textit{decidable} to determine whether or not $L$ is \textit{sdd-closed}

For a given context-free languages $L$,
- it is \textit{undecidable} to determine whether or not $L$ is \textit{sdi-closed}
- it is \textit{undecidable} to determine whether or not $L$ is \textit{sdd-closed}
For a given context-free language $L$, it is **undecidable** to determine whether or not $L$ is sdi-closed.

We use a reduction from the *Post Correspondence Problem* (PCP)

- Let $((u_1, u_2, \ldots, u_n), (v_1, v_2, \ldots, v_n))$ be an instance of PCP, where $u_i, v_i \in \Sigma^*$ and $1 \leq i \leq n$
- A solution of the PCP instance is $i_1, \ldots, i_k \in \{1, \ldots, n\}$ such that

$$u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$$
**Sdi-Closed of Site-Directed Insertion**

**Theorem**

For a given context-free language $L$, it is **undecidable** to determine whether or not $L$ is sdi-closed.

We use a reduction from the *Post Correspondence Problem* (PCP)

**Example**

Let $I_{PCP} = ((ab, bbb, a)(ab, b, bba))$.

The solution is $2, 3, 1$ since

$$u_2 u_3 u_1 = v_2 v_3 v_1 = bbbaab.$$
Theorem
For a given context-free language $L$, it is undecidable to determine whether or not $L$ is sdi-closed.

We use a reduction from the Post Correspondence Problem (PCP)

- Let $((u_1, u_2, \ldots, u_n), (v_1, v_2, \ldots, v_n))$ be an instance of PCP, where $u_i, v_i \in \Sigma^*$ and $1 \leq i \leq n$
- A solution of this instance is $i_1, \ldots, i_k \in \{1, \ldots, n\}$ such that
  \[
  u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}
  \]
- PCP is undecidable!
Theorem

For a given context-free language $L$, it is undecidable to determine whether or not $L$ is sdi-closed.

Idea

- We define $L = L_1 \cup L_2$ which is context-free and
- claim that “PCP has a solution iff $L$ is not sdi-closed".
**Sdi-Closed of Site-Directed Insertion**

\[
L_1 = \{ \$i_1 i_2 \cdots i_r \# u_{i_r}^R u_{i_{r-1}}^R \cdots u_{i_1}^R \# \# v_{j_1} v_{j_2} \cdots v_{j_s} \# j_s j_{s-1} \cdots j_1 \$ \mid r, s \geq 1, 1 \leq i_x, j_y \leq n, 1 \leq x \leq r, 1 \leq y \leq s \}
\]

\[
L_2 = \{ \$i_1 i_2 \cdots i_r \# w \# f \# w^R \# i_r i_{r-1} \cdots i_1 \$ \mid w \in \{a, b\}^*, r \geq 1, 1 \leq i_x \leq n, 1 \leq x \leq r \}
\]

Suppose \((i_1, \ldots, i_k)\) is PCP solution

\[
\begin{array}{|c|c|}
\hline
\text{\$i_1 i_2 \cdots i_k \# u_{i_k}^R u_{i_{k-1}} \cdots u_{i_1}^R \# \# v_{i_1} v_{i_2} \cdots v_{i_k} \# i_k i_{k-1} \cdots i_1 \$} & w_1 \in L_1 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|}
\hline
\text{\$i_1 i_2 \cdots i_k \# u_{i_k}^R u_{i_{k-1}} \cdots u_{i_1}^R \# f \# v_{i_1} v_{i_2} \cdots v_{i_k} \# i_k i_{k-1} \cdots i_1 \$} & w_2 \in L_2 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|}
\hline
\text{\$i_1 i_2 \cdots i_k \# u_{i_k}^R u_{i_{k-1}} \cdots u_{i_1}^R \# f \# v_{i_1} v_{i_2} \cdots v_{i_k} \# i_k i_{k-1} \cdots i_1 \$} & \in w_1 \leftarrow w_2 \\
\hline
\end{array}
\]
Outline

1 Introduction
   - Background
   - Definition
   - Problems

2 Main Results
   - Closure Properties
   - Decidability

3 Conclusion
   - Summary
   - Future Works
Summary

- Regular languages are closed under $\text{sdi} \leftrightarrow$ and $\text{sdd} \leftrightarrow$, whereas context-free languages are not.
- $\text{SDI}^*(L)$ is not regular when $L$ is finite.
- $\text{SDD}^*(L)$ is not context-free when $L$ is context-free.
- $\text{sdi}(\text{sdd})$-closeness: decidable when $L$ is regular
- $\text{sdi}(\text{sdd})$-closeness: undecidable when $L$ is context-free
- We can decide whether or not $x \xleftarrow{\text{sdd}} y \neq \emptyset$ and $z \in x \xleftarrow{\text{sdd}} y$ in linear time.
1 Introduction
- Background
- Definition
- Problems

2 Main Results
- Closure Properties
- Decidability

3 Conclusion
- Summary
- Future Works
Open Problems and Future Works

- Closure property under $\text{SDD}^*$ and $\text{SDI}^*$
- Properties of maximal/minimal site-directed insertion/deletion
Thank you for your attention!