ソースコードの構文木表現による
構造類似性を用いたプログラム作成支援方式

北 椋太　北川 高嗣　中西 崇文　岡田 龍太郎
武蔵野大学データサイエンス学部　TransMedia Tech Lab

目次

- 研究背景
- 研究目的
- 提案方式
- まとめ
- 今後の展望

研究背景

- プログラムの生産性・保守性の向上が課題
- GitHubなどのオープンプラットフォームにより、
  インターネット上にプログラムが散在し、共有されている

研究目的

ソースコードの構文木表現による
構造類似性を用いたプログラム作成支援方式の実現

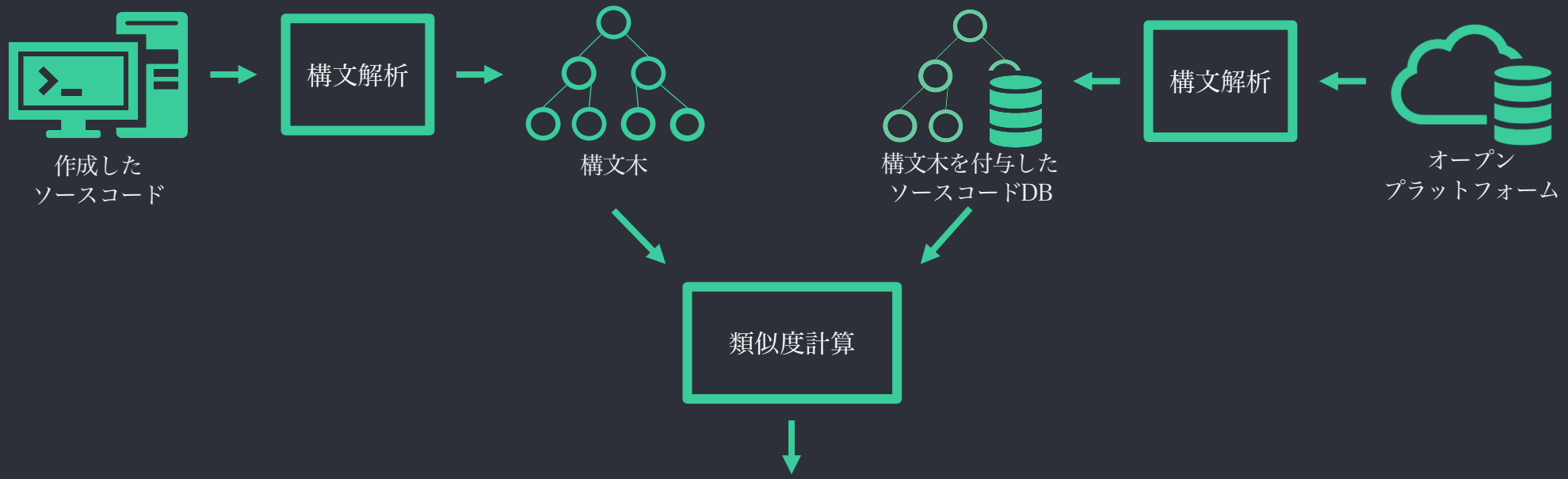
類似構造をもつソースコードをデータベースから検索可能

作成中のプログラムのソースコードと類似した機能を持つソースコードを引用することにより、

初学者のプログラミングスキルの向上につながる。

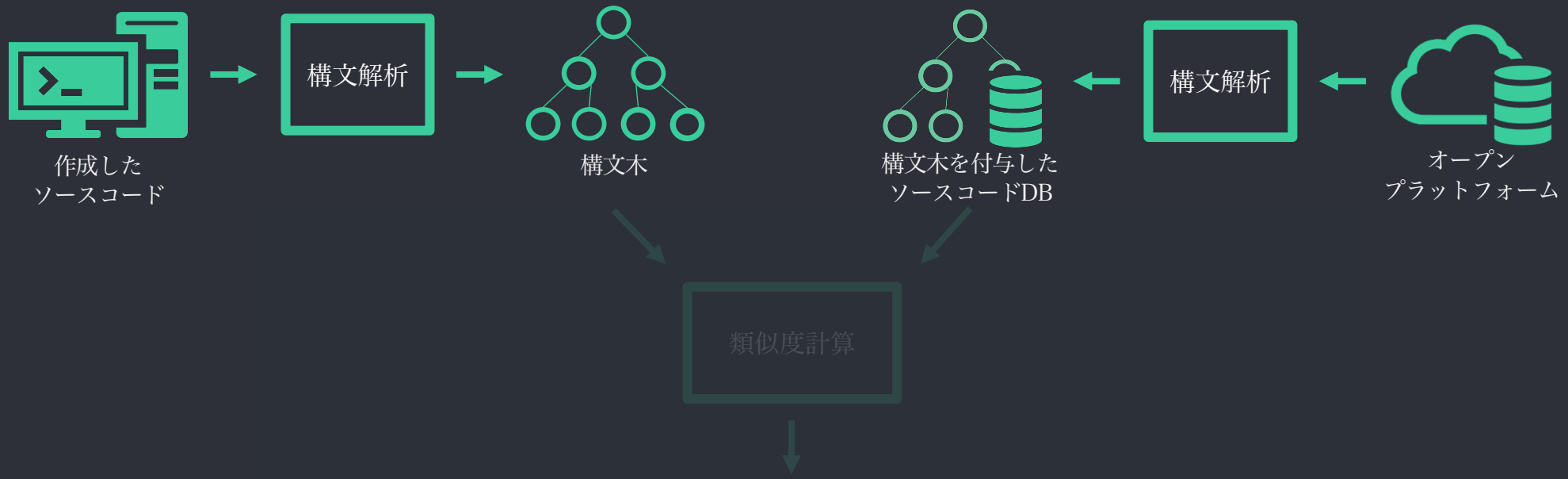
また、同機能をもつソースコードのパターンを統一することにより、保守性が向上する。

提案方式 – システム全体像 –



作成したソースコードと類似した構造を持つ外部のソースコードを提示

提案方式 – 構文解析 –



作成したソースコードと類似した構造を持つ外部のソースコードを提示

提案方式 – 構文解析 –

ソースコード

↓

字句解析

↓

構文解析

↓

構文木

```
def HelloWorld():
    print("Hello world!")

HelloWorld()
```

ソースコード

↓

```
[TokenInfo(type=62 (ENCODING), string='utf-8', start=(0, 0), end=(0, 0), line=''),
 TokenInfo(type=1 (NAME), string='def', start=(1, 0), end=(1, 3), line='def ...
 TokenInfo(type=1 (NAME), string='HelloWorld', start=(1, 4), end=(1, 14), line= ...
 TokenInfo(type=54 (OP), string='(', start=(1, 14), end=(1, 15), line='def ...
 TokenInfo(type=54 (OP), string=')', start=(1, 15), end=(1, 16), line='def ...
 TokenInfo(type=54 (OP), string=':', start=(1, 16), end=(1, 17), line='def ...
 TokenInfo(type=4 (NEWLINE), string='¥n', start=(1, 17), end=(1, 18), line='def ...
                                ⋮
```

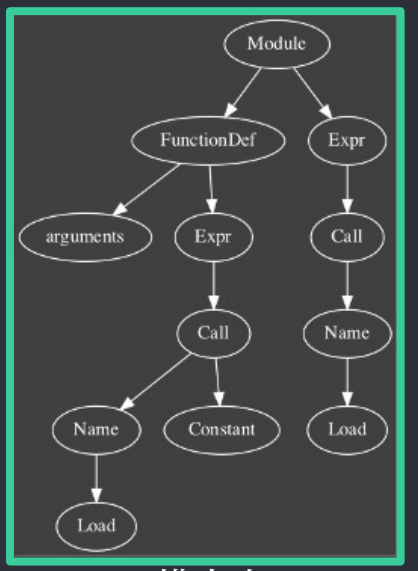
字句解析

↓

```
<class '_ast.Module'>
    <class '_ast.Expr'>: 4
        <class '_ast.Call'>: 4
            <class '_ast.Name'>: 4
                <class '_ast.Load'>
                                ⋮
```

構文解析

→



構文木

字句解析

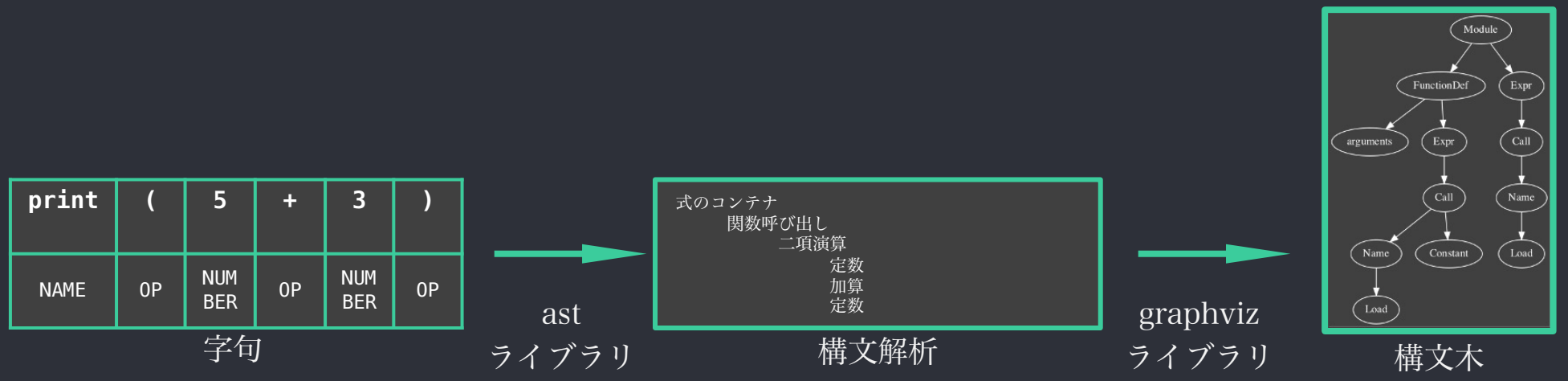
字句　　... プログラム内における文字列の最小単位

>> 自然言語における単語

字句解析　... ソースプログラムを字句単位に分割すること

構文解析

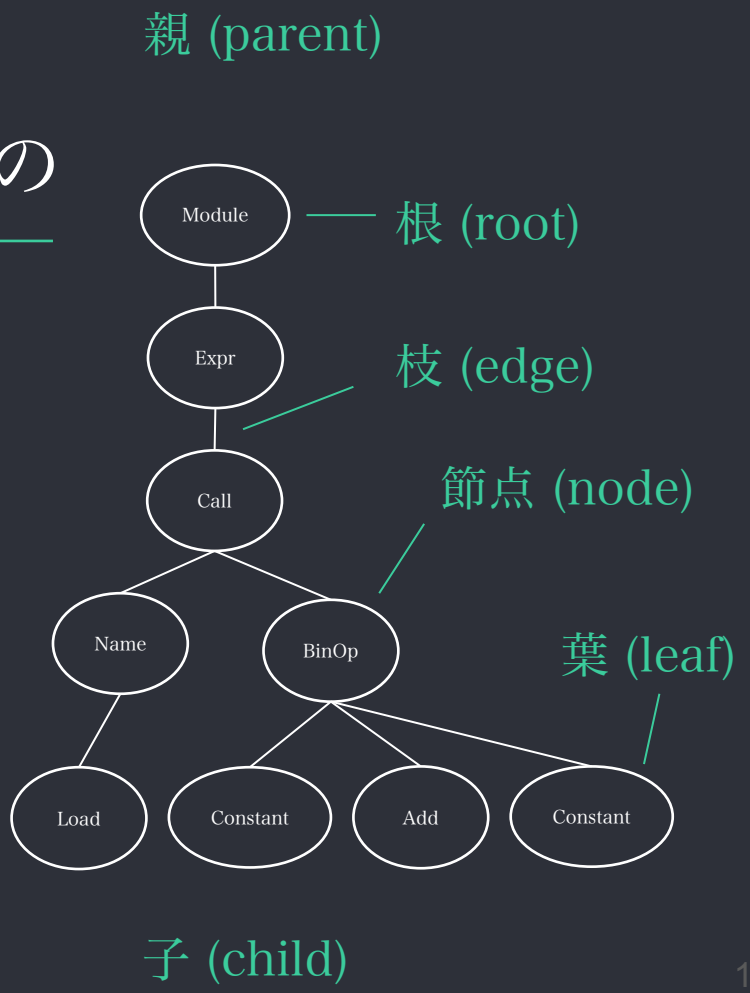
構文解析　... 字句解析によって受渡された字句を構文規則に従って関係性を解析し木構造を生成する



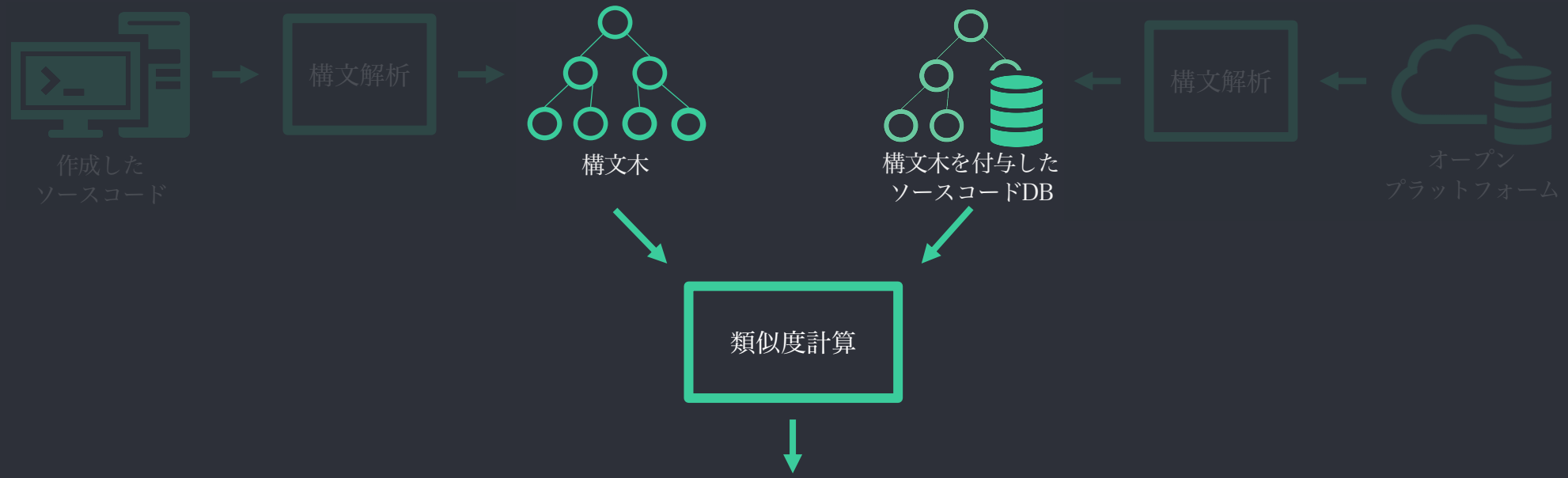
構文木

構文木 … 構文解析の結果を木構造で表したもの

木構造… 閉路を持たない連結されたグラフ



提案方式 – 類似度計算 –



作成したソースコードと類似した構造を持つ外部のソースコードを提示

提案方式 – 類似度計算 –

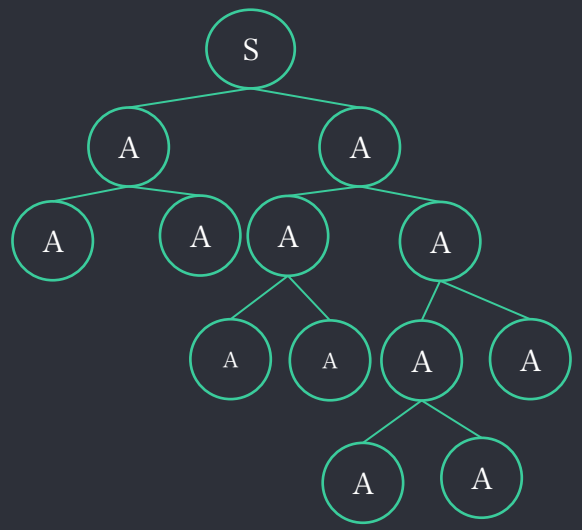
部分木の一致度による類似度計算

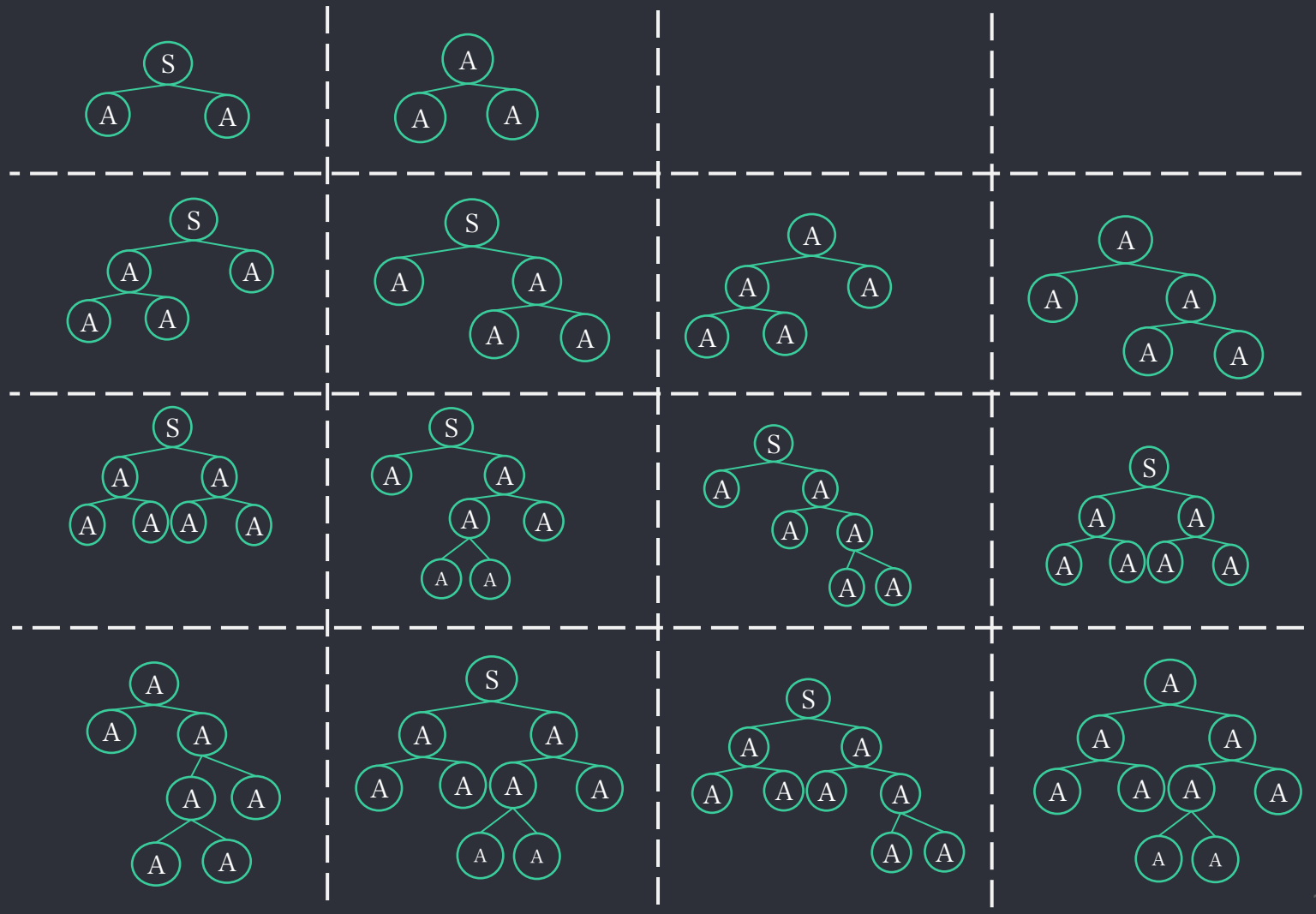
1. 2つの構文木から部分木集合をそれぞれ作成する。
2. 作成した2つの部分木集合のうち、一致する個数を計算

椎名広光, 秋友克俊. 「構文解析木の類似度の判定アルゴリズム」.数理解析研究所講究録 1426 巻 2005 年 26-31
https://www.kurims.kyoto-u.ac.jp/~kyodo/kokyuroku/contents/pdf/1426-5.pdf

部分木の抽出

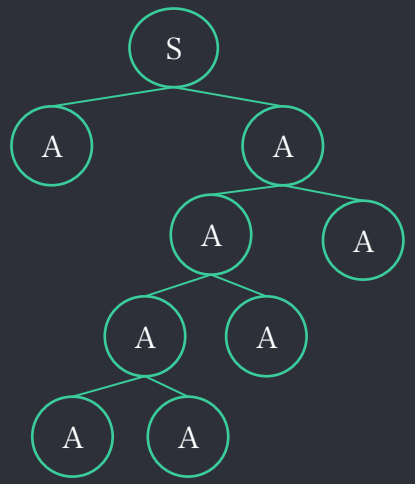
T_A

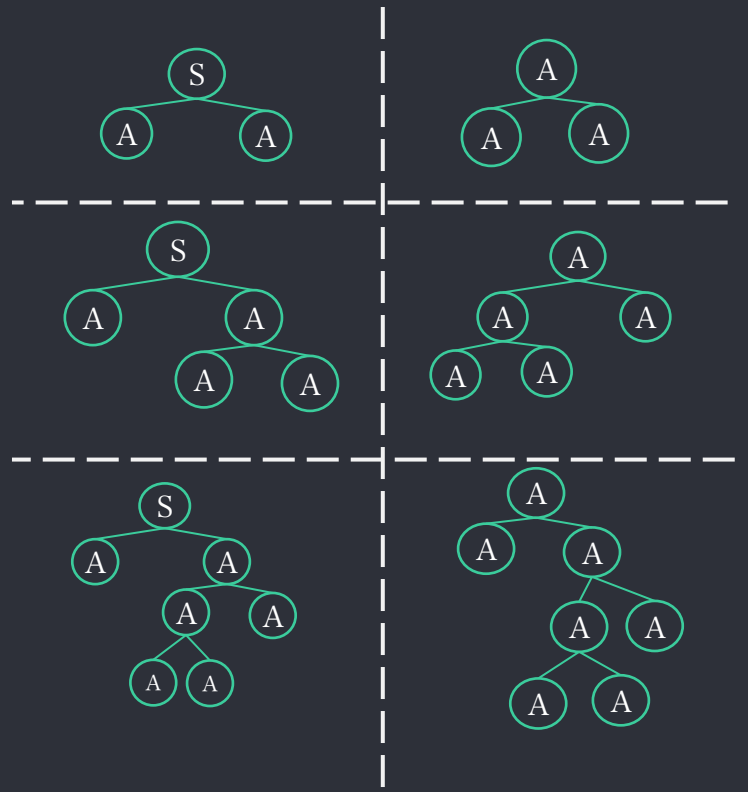

S
A A
A A A A
A A A A
A A


S
A A
A
A A
S
A A
A A
S
A A
A A
A
A A
A A
A
A A
A A
S
A A
A A
S
A A
A A
S
A A
A A
A A
S
A A
A A
A A
A
A A
A A
A A
S
A A
A A
A A
A
A A
A A
A A
A
A A
A A
A A
S
A A
A A
A A
A
A A
A A
A A
A
A A
A A
A A
A
A A
A A
A A
A
A A
A A

部分木の抽出

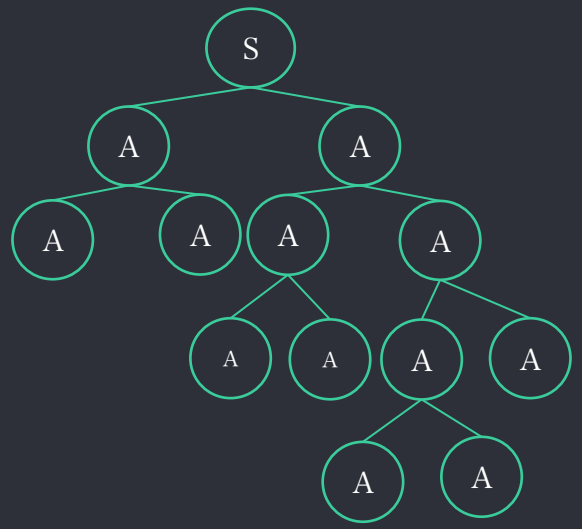
T_{B}



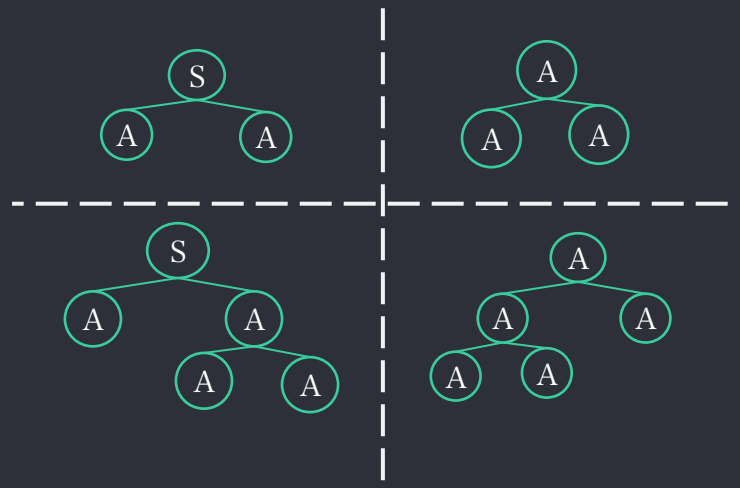


部分木の抽出

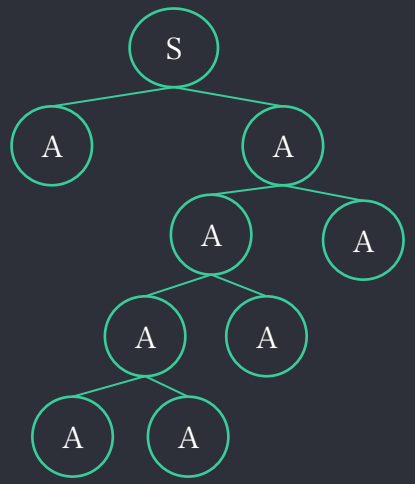
T_A



共通する部分木

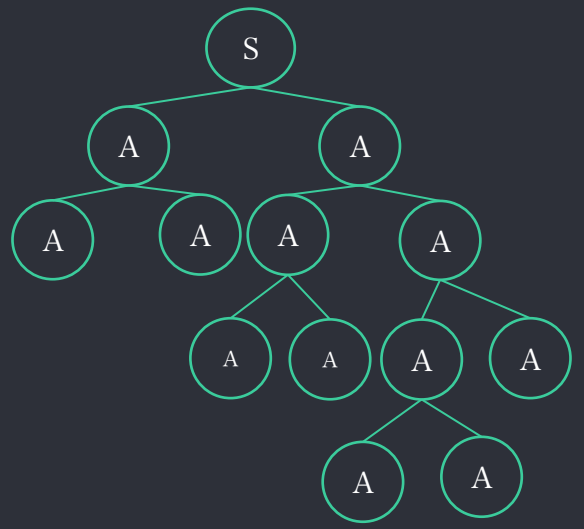


T_B



部分木の一致度による類似度計算

T_A

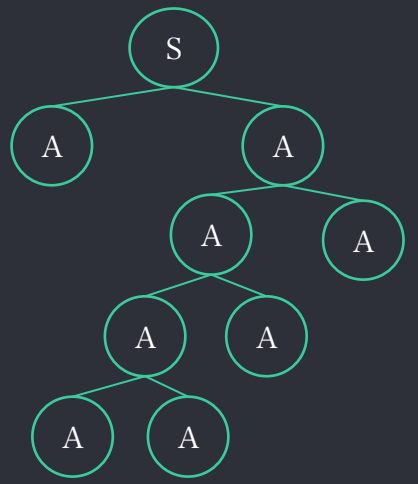


$$sim(T_A, T_B) = \frac{2}{N_A + N_B} \sum_{k=1}^{N_A} \sum_{l=1}^{N_B} com(\alpha_k, \beta_l)$$

$$com(T_A, T_B) = \begin{cases} 1, \alpha_k = \beta_l \\ 0, \alpha_k \neq \beta_l \end{cases}$$

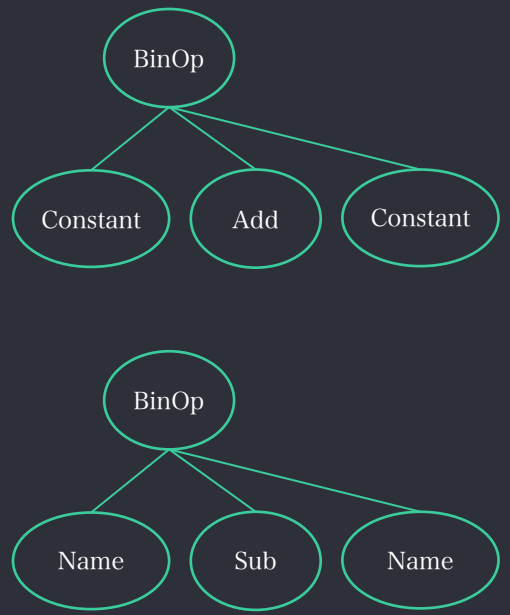
$$sim(T_A, T_B) = \frac{2}{14+6} \cdot 4 = \underline{0.4}$$

T_B



部分木の一致度による類似度計算

本研究の特徴：nodeの意味を考慮して部分木を抽出

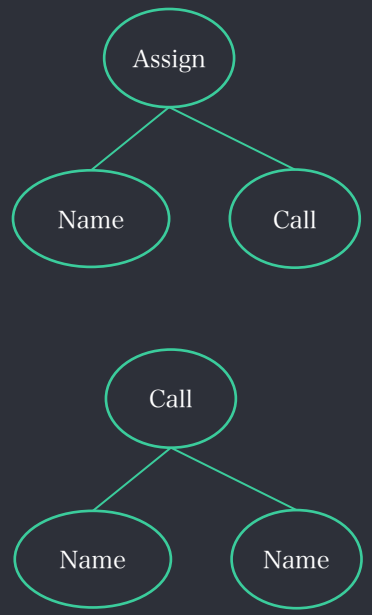


意味的差異はない

>> 同一の部分木とみなす

部分木の一致度による類似度計算

本研究の特徴：nodeの意味を考慮して部分木を抽出



意味的差異がある

>> 不同の部分木とみなす

まとめ

- ソースコードの構文木表現による構造類似性を用いたプログラム作成支援方式を提案した
- 類似構造をサジェストすることで保守性の向上を図る
- 本研究の特徴として、
  ノードの意味的特徴に注目し、部分木の抽出を行う

今後の展望

- 部分木の抽出条件を設定
- 編集距離を用いた類似度計算等との比較

- 同一ソースコード内の類似構造から自動で関数化
- コメントとプログラムの対応構造からdocstringの自動生成