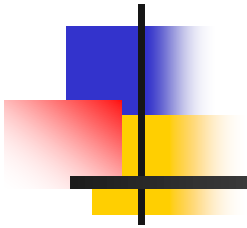
形式概念解析を用いた密部分抽出



京都大学 情報学研究科
山本　章博

形式概念解析の応用例(1)

オープン・ソフトウェア共同開発における
メーリングリストを用いたソースファイル間の
関係の抽出

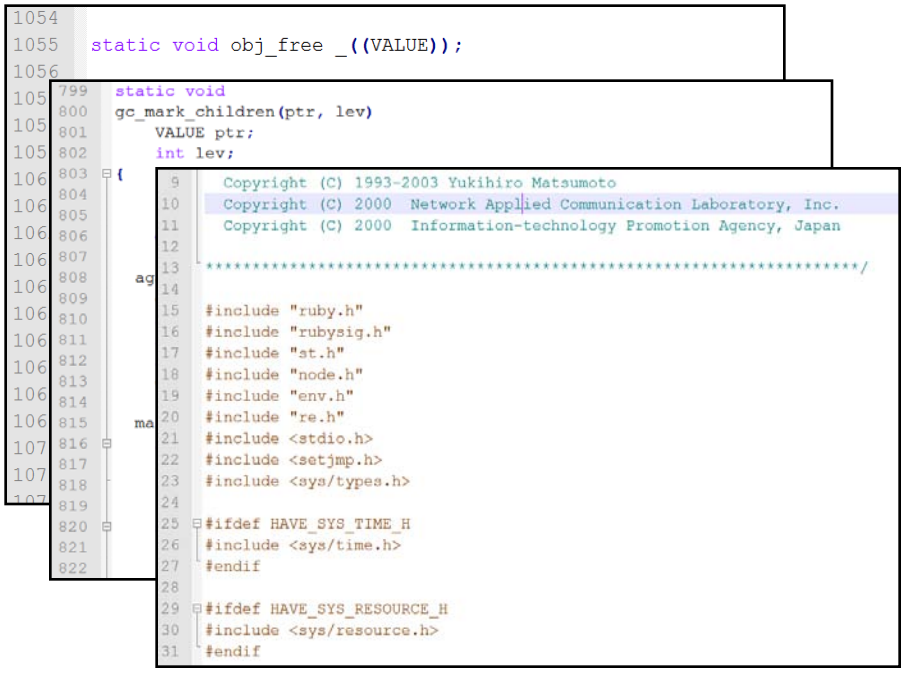
Dinh et al.: Discovering the Structures of Open Source Programs from Their Developer Mailing Lists, 2009

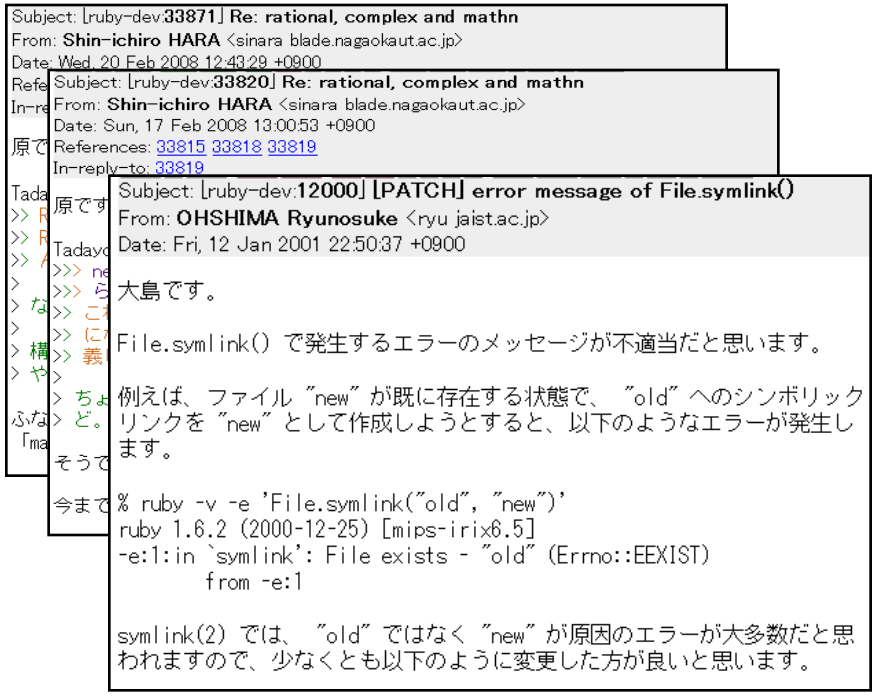
- 複雑化してしまうオープン・ソフトウェアのソースファイルを管理
  - 仕様書，ソースファイル間の参照関係を用いない

メーリングリストの利用

- 開発者間のメール連絡の記録とソースファイルとの関係を抽出

ソース・ファイル ⟺ 開発者メーリングリスト





メール・ソース間の参照関係

- メール本文中から参照関係

関係1：モジュール名が件名に出現
関係2：モジュール名が本体のテキスト部分に出現
関係3：モジュール名が本体のコード部分に出現
関係4：モジュールの一部が本体に引用

ソースファイル

```
#xyz.h

void read_input_html()
…
…
while(temp<size){
   process_data(&input_array);
   count++;
}
…
…
```

メール

Sub:　ファイル**xyz.h**の修正点

… read_input_htmlはHTMLファイルを読み込む関数である. …

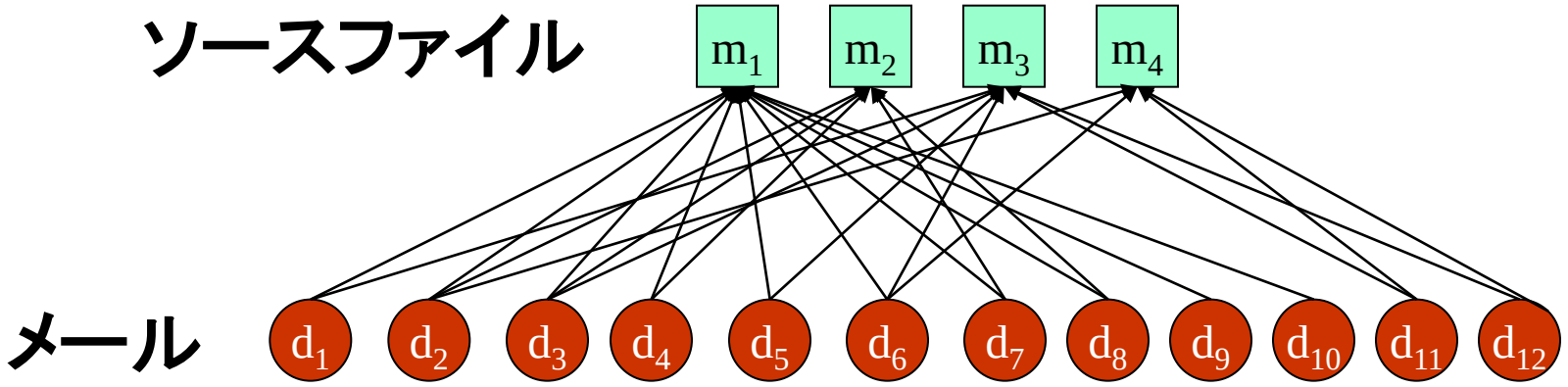
```
…
#include ”xyz.h”
…
```

…
ファイル**xyz.h**を次のように修正しまた.
…

```
…
while(temp<size){
   process_data(&input_array);
   count++;
}
…
```

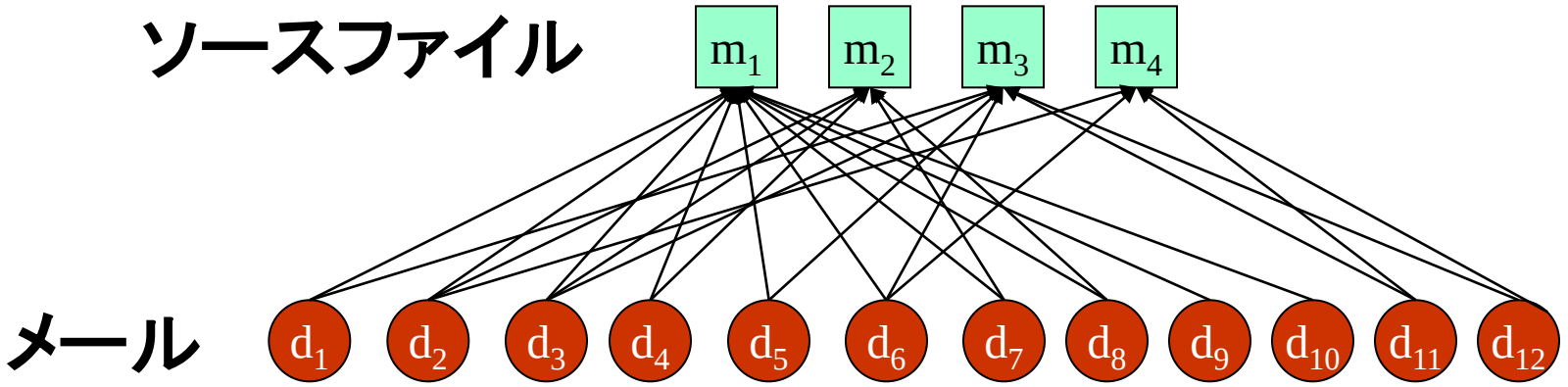
2部グラフの作成と形式概念

- ソースファイルとメールの間の参照関係を2部グラフで表現



形式概念：2部グラフの意味での極大閉グラフ

2部グラフとコンテクスト表



コンテクスト表

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
m_1	●	●	●	●	●	●	●	●	●	●		
m_2	●	●	●	●			●	●				
m_3	●	●	●		●	●					●	●
m_4	●	●		●	●	●					●	●

コンテクスト表と形式概念解析

形式概念：コンテキスト表において，行と列の順序を適当に並べ替えたとき，極大になる矩形領域

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
m_1	●	●	●	●	●	●	●	●	●	●		
m_2	●	●	●	●			●	●				
m_3	●	●	●		●	●					●	●
m_4	●	●		●	●	●					●	●

	d_1	d_2	d_3	d_4	d_7	d_8	d_5	d_6	d_9	d_{10}	d_{11}	d_{12}
m_1	●	●	●	●	●	●	●	●	●	●		
m_2	●	●	●	●	●	●						
m_3	●	●	●				●	●			●	●
m_4	●	●		●			●	●			●	●

形式概念間の包摂関係

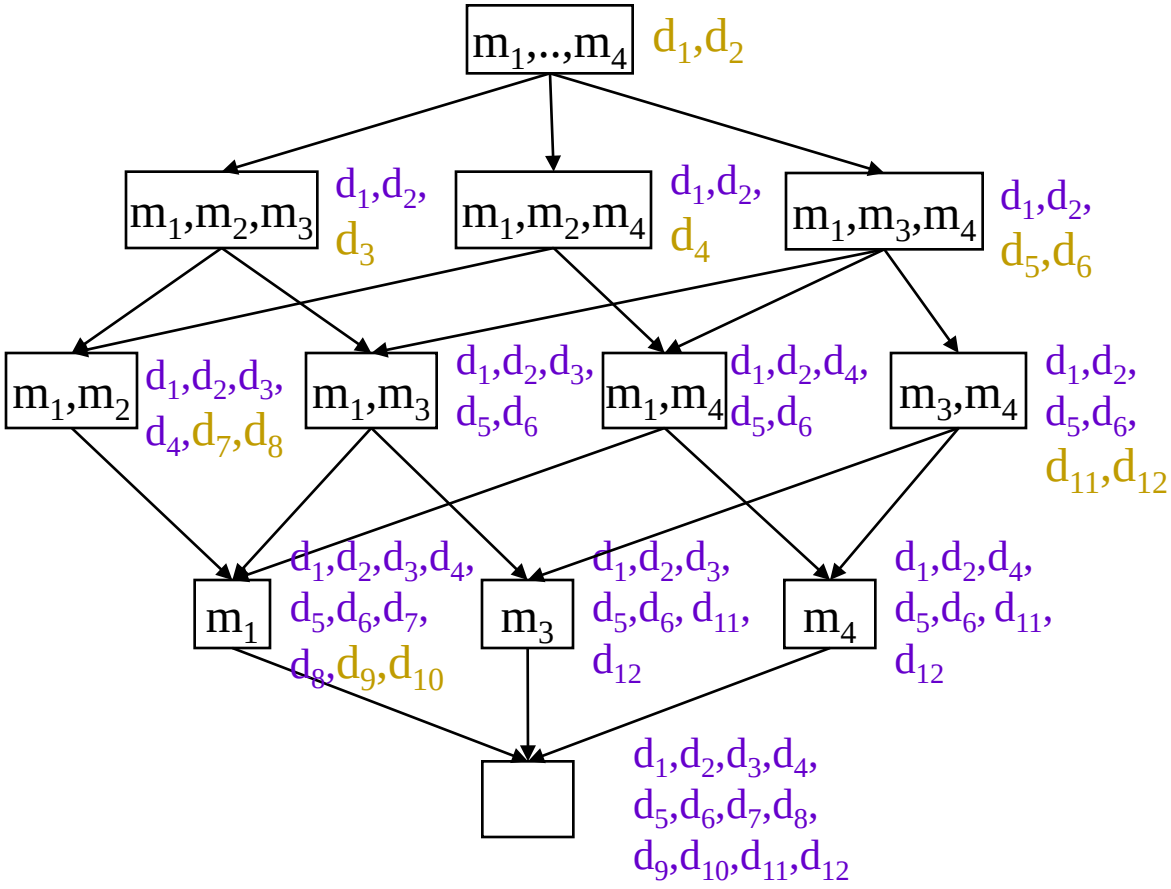
- 形式概念間にも包摂関係がある
  - ソースファイルが多い ⇔ メールが少ない

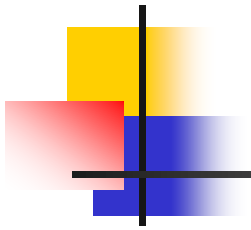
	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
m_1	●	●	●	●	●	●	●	●	●	●		
m_2	●	●	●	●			●	●				
m_3	●	●	●		●	●					●	●
m_4	●	●		●	●	●					●	●

	d_1	d_2	d_3	d_4	d_7	d_8	d_5	d_6	d_9	d_{10}	d_{11}	d_{12}
m_1	●	●	●	●	●	●	●	●	●	●		
m_2	●	●	●	●	●	●						
m_3	●	●	●				●	●			●	●
m_4	●	●		●			●	●			●	●

形式概念間の包摂関係

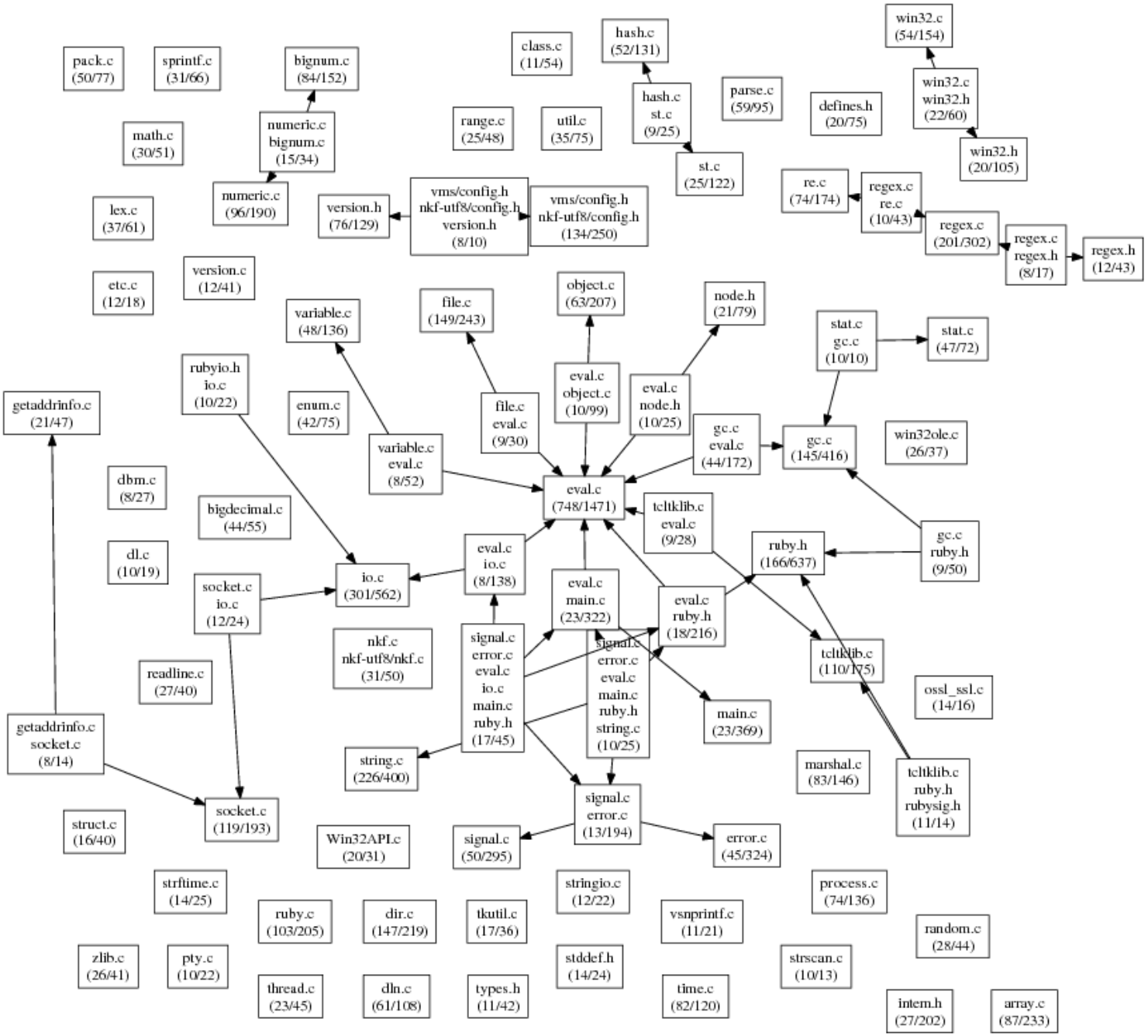
- 形式概念間の包摂関係をグラフ(Hasse図)で表示

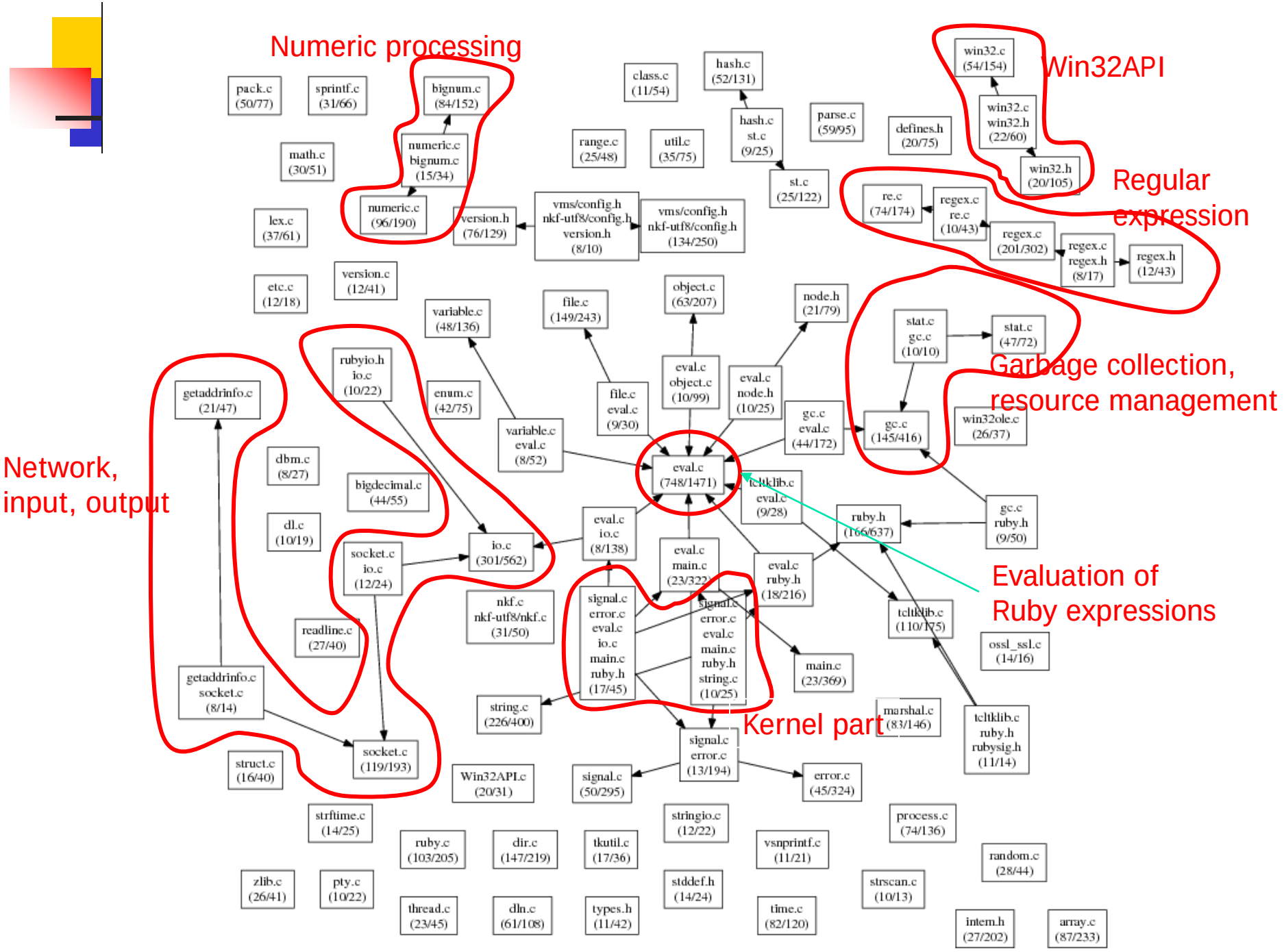




有用なHasse図にするために

- 参照関係の抽出
- 包摂関係(Hasse図)の枝刈り


pack.c (50/77)
sprintf.c (31/66)
bignum.c (84/152)
class.c (11/54)
hash.c (52/131)
win32.c (54/154)
math.c (30/51)
numeric.c bignum.c (15/34)
range.c (25/48)
util.c (35/75)
hash.c st.c (9/25)
parse.c (59/95)
defines.h (20/75)
win32.c win32.h (22/60)
win32.h (20/105)
numeric.c (96/190)
st.c (25/122)
lex.c (37/61)
version.h (76/129)
vms/config.h nkf-utf8/config.h version.h (8/10)
vms/config.h nkf-utf8/config.h (134/250)
re.c (74/174)
regex.c re.c (10/43)
regex.c (201/302)
regex.c regex.h (8/17)
regex.h (12/43)
etc.c (12/18)
version.c (12/41)
variable.c (48/136)
file.c (149/243)
object.c (63/207)
node.h (21/79)
stat.c gc.c (10/10)
stat.c (47/72)
rubyio.h io.c (10/22)
eval.c object.c (10/99)
eval.c node.h (10/25)
getaddrinfo.c (21/47)
enum.c (42/75)
file.c eval.c (9/30)
gc.c eval.c (44/172)
gc.c (145/416)
win32ole.c (26/37)
variable.c eval.c (8/52)
dbm.c (8/27)
eval.c (748/1471)
tcltklib.c eval.c (9/28)
bigdecimal.c (44/55)
ruby.h (166/637)
gc.c ruby.h (9/50)
dl.c (10/19)
eval.c io.c (8/138)
io.c (301/562)
eval.c main.c (23/322)
eval.c ruby.h (18/216)
socket.c io.c (12/24)
tcltklib.c (110/175)
ossl_ssl.c (14/16)
nkf.c nkf-utf8/nkf.c (31/50)
signal.c error.c eval.c io.c main.c ruby.h (17/45)
signal.c error.c eval.c main.c ruby.h string.c (10/25)
readline.c (27/40)
main.c (23/369)
getaddrinfo.c socket.c (8/14)
string.c (226/400)
marshal.c (83/146)
tcltklib.c ruby.h rubysig.h (11/14)
socket.c (119/193)
struct.c (16/40)
signal.c error.c (13/194)
Win32API.c (20/31)
signal.c (50/295)
error.c (45/324)
strftime.c (14/25)
stringio.c (12/22)
process.c (74/136)
ruby.c (103/205)
dir.c (147/219)
tkutil.c (17/36)
vsnprintf.c (11/21)
random.c (28/44)
zlib.c (26/41)
pty.c (10/22)
stddef.h (14/24)
strscan.c (10/13)
thread.c (23/45)
dln.c (61/108)
types.h (11/42)
time.c (82/120)
intern.h (27/202)
array.c (87/233)


Numeric processing
Win32API
Regular expression
Garbage collection, resource management
Network, input, output
Evaluation of Ruby expressions
Kernel part
pack.c (50/77)
sprintf.c (31/66)
bignum.c (84/152)
numeric.c bignum.c (15/34)
numeric.c (96/190)
math.c (30/51)
lex.c (37/61)
version.h (76/129)
vms/config.h nkf-utf8/config.h version.h (8/10)
vms/config.h nkf-utf8/config.h (134/250)
version.c (12/41)
etc.c (12/18)
class.c (11/54)
hash.c (52/131)
hash.c st.c (9/25)
st.c (25/122)
range.c (25/48)
util.c (35/75)
parse.c (59/95)
defines.h (20/75)
win32.c (54/154)
win32.c win32.h (22/60)
win32.h (20/105)
re.c (74/174)
regex.c re.c (10/43)
regex.c (201/302)
regex.c regex.h (8/17)
regex.h (12/43)
variable.c (48/136)
file.c (149/243)
object.c (63/207)
node.h (21/79)
stat.c gc.c (10/10)
stat.c (47/72)
rubyio.h io.c (10/22)
enum.c (42/75)
file.c eval.c (9/30)
eval.c object.c (10/99)
eval.c node.h (10/25)
gc.c eval.c (44/172)
gc.c (145/416)
win32ole.c (26/37)
getaddrinfo.c (21/47)
variable.c eval.c (8/52)
eval.c (748/1471)
tcltklib.c eval.c (9/28)
gc.c ruby.h (9/50)
dbm.c (8/27)
bigdecimal.c (44/55)
dl.c (10/19)
eval.c io.c (8/138)
ruby.h (166/637)
socket.c io.c (12/24)
io.c (301/562)
eval.c main.c (23/322)
eval.c ruby.h (18/216)
tcltklib.c (110/175)
ossl_ssl.c (14/16)
readline.c (27/40)
nkf.c nkf-utf8/nkf.c (31/50)
signal.c error.c eval.c io.c main.c ruby.h (17/45)
signal.c error.c eval.c main.c ruby.h string.c (10/25)
main.c (23/369)
getaddrinfo.c socket.c (8/14)
string.c (226/400)
marshal.c (83/146)
tcltklib.c ruby.h rubysig.h (11/14)
socket.c (119/193)
struct.c (16/40)
Win32API.c (20/31)
signal.c (50/295)
signal.c error.c (13/194)
error.c (45/324)
strftime.c (14/25)
stringio.c (12/22)
process.c (74/136)
ruby.c (103/205)
dir.c (147/219)
tkutil.c (17/36)
vsnprintf.c (11/21)
random.c (28/44)
zlib.c (26/41)
pty.c (10/22)
stddef.h (14/24)
strscan.c (10/13)
thread.c (23/45)
dln.c (61/108)
types.h (11/42)
time.c (82/120)
intern.h (27/202)
array.c (87/233)

参照関係抽出の実際

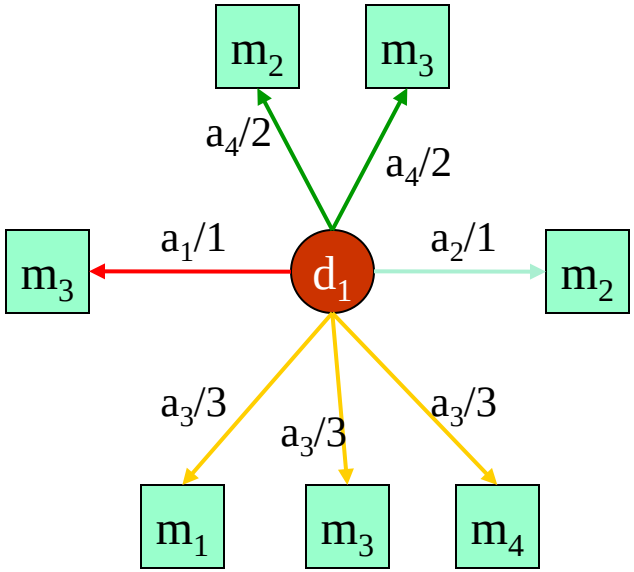
- サポート集合R_Sを求める
  - $R_S = \{(d_i, m_j) \in D \times M \mid \exists k.\ e_k(d_i, m_j) = 1\}$
- R_Sをフィルタリングし, R*を求める
  - $R^* = \{(d_i, m_j) \in R_S \mid conf(d_i, m_j) > 0\}$
  - ここで,

$$conf(d_i, m_j) = b + \sum_{k=1}^{4} a_k \frac{e_k(d_i, m_j)}{\#n_{e_k}(d_i)}$$

証拠kが見つかった

d_iと関連証拠kが見つかったモジュールの数

フィルタリングの例



メールd_1と関連する証拠

証拠1：モジュール名が件名に出現
証拠2：モジュール名が本体のテキスト部分に出現
証拠3：モジュール名が本体のコード部分に出現
証拠4：モジュールの一部が本体に引用

パラメータの値（例）：

$$a_1 = a_2 = a_3 = a_4 = \frac{1}{4}, b = -\frac{1}{4}$$

（実際には実験データから推定される）

関連性の信頼度：

$$conf(d_1, m_1) = b + \frac{a_3}{3} = -\frac{1}{6}$$

$$conf(d_1, m_2) = b + \frac{a_2}{1} + \frac{a_4}{2} = \frac{1}{8}$$

$$conf(d_1, m_3) = b + \frac{a_1}{1} + \frac{a_3}{3} + \frac{a_4}{2} = \frac{5}{24}$$

$$conf(d_1, m_4) = b + \frac{a_3}{3} = -\frac{1}{6}$$

実験に用いたデータセット

	開発言語	共同開発時期	メール数（日本語）	ファイル数
HOS[1]	C	2002年～	1513	186
Namazu[2]	C,Perl	1997年～	9462	164
Ruby[3]	C	1997年～	36399	255

[1] HOS: *Hyper Operating System*
[2] Namazu 日本語全文検索システム
[3] Ruby プログラミング言語

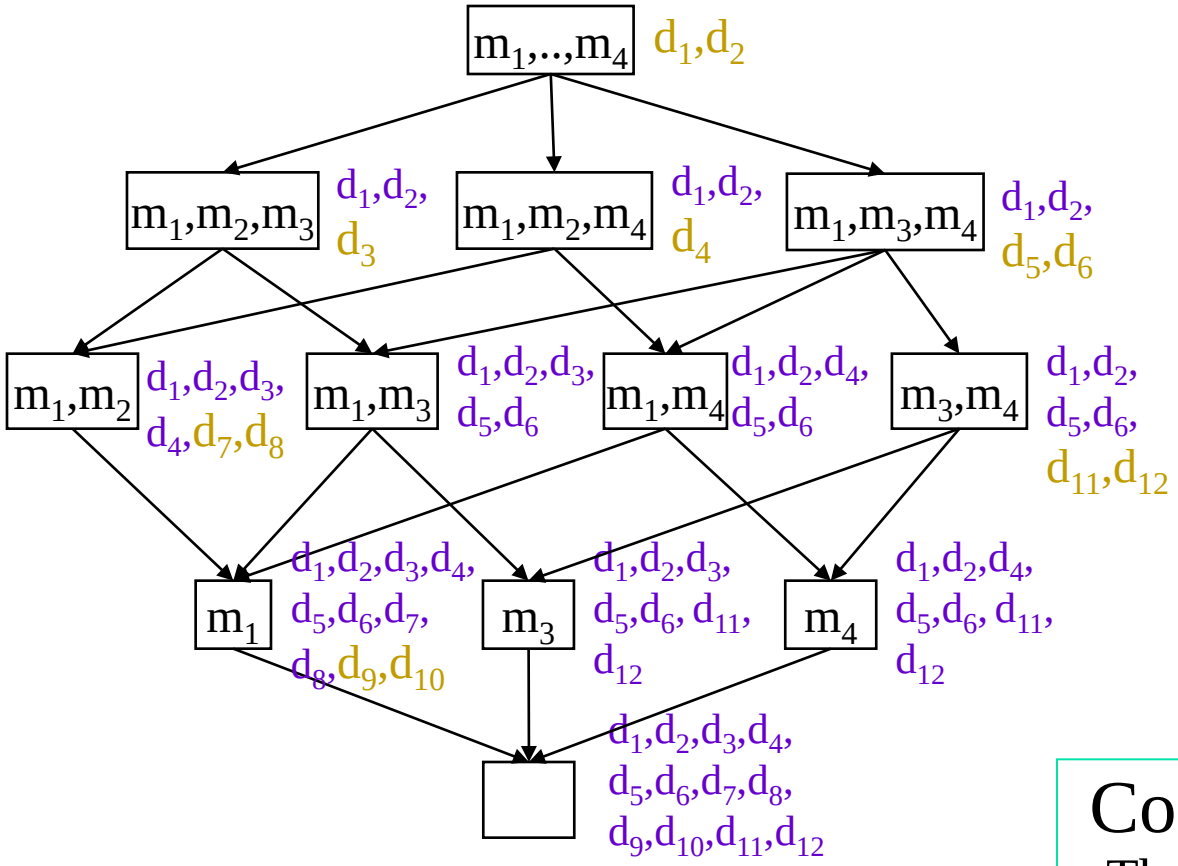
参照関係抽出実験結果

Project	Data	Supported		Filtered	
	$\frac{\lvert\mathcal{D}\rvert}{\lvert\mathcal{M}\rvert}$	$\frac{\lvert\mathcal{D}_S\rvert}{\lvert\mathcal{M}\rvert}$	$\frac{\lvert\mathcal{R}_S\rvert}{\lvert\mathcal{M}\rvert}$	$\frac{\lvert\mathcal{D}^*\rvert}{\lvert\mathcal{M}\rvert}$	$\frac{\lvert\mathcal{R}^*\rvert}{\lvert\mathcal{M}\rvert}$
HOS	8.2	2.0	7.2	1.7	5.3
Namazu	57.8	15.05	93.3	9.5	33.1
Ruby	142.7	30.1	111.8	25.7	89.8

- 約20%のメールがいずれかのモジュールと関連性を持つメールとして抽出可能
- Namazuの集合R_Sの約10%に注釈を付加した．回帰分析によるパラメータ推定結果：

a_1=0.141, a_2=0.459, a_3=0.373, a_4=0.027, b= -0.033

Hasse図の枝刈り

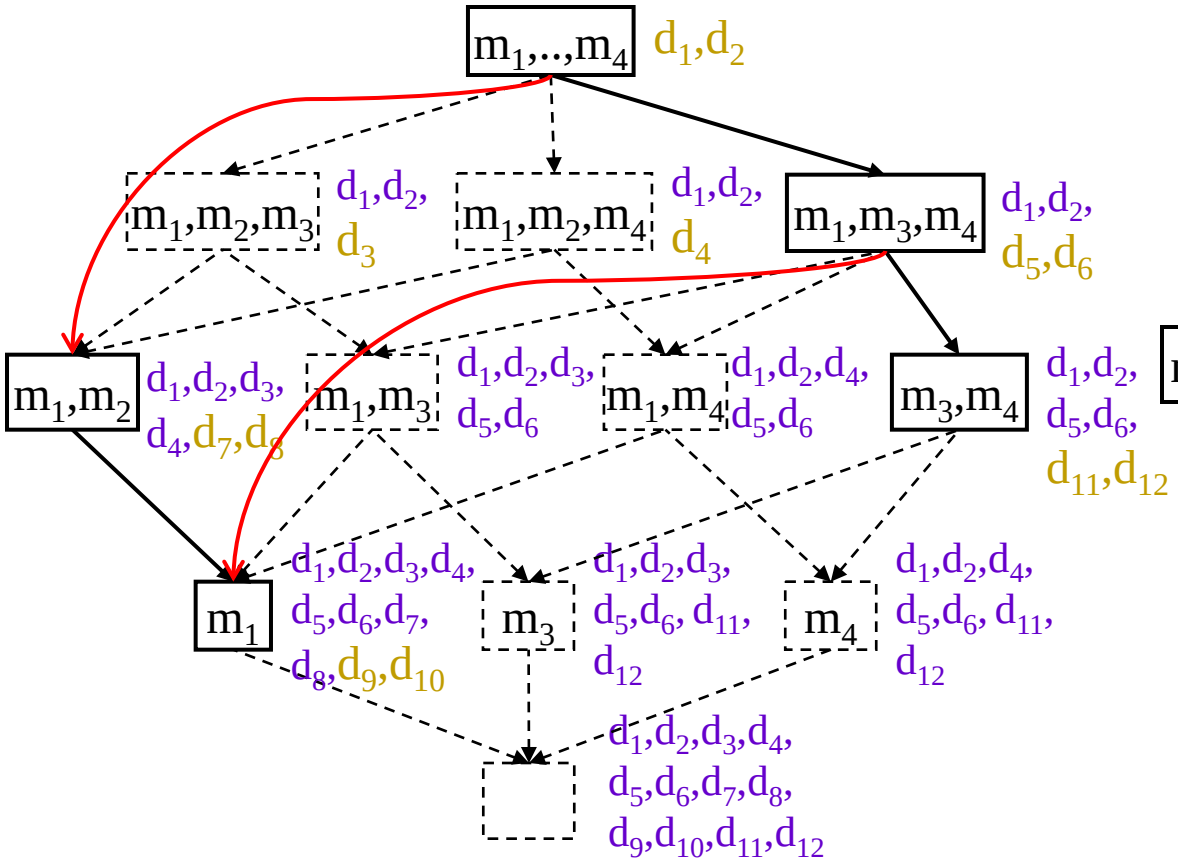


Lattice before pruning

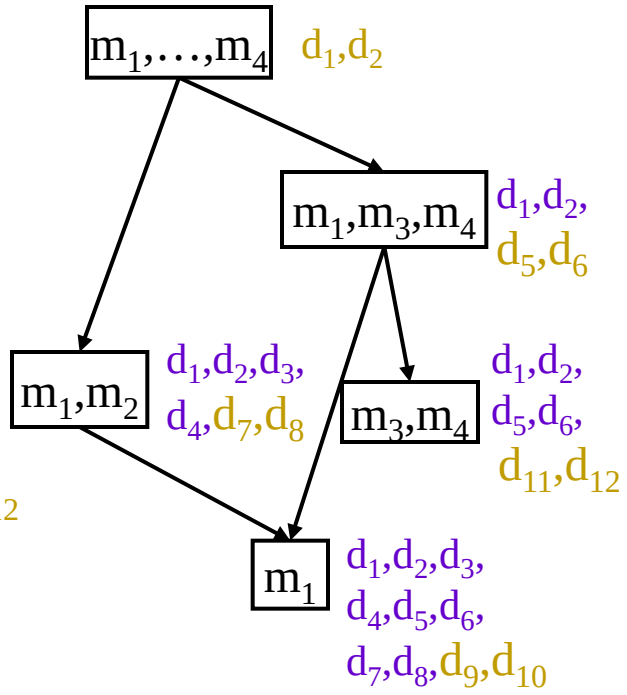
Constraints:

- The number of its possessed attributes $\geqq \sigma$
- The number of its introduced attributes $\geqq \tau$

Hasse図の枝刈り



Lattice during pruning

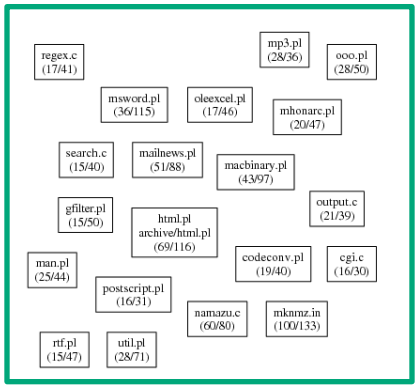


Lattice after pruning
（Extracted structure）

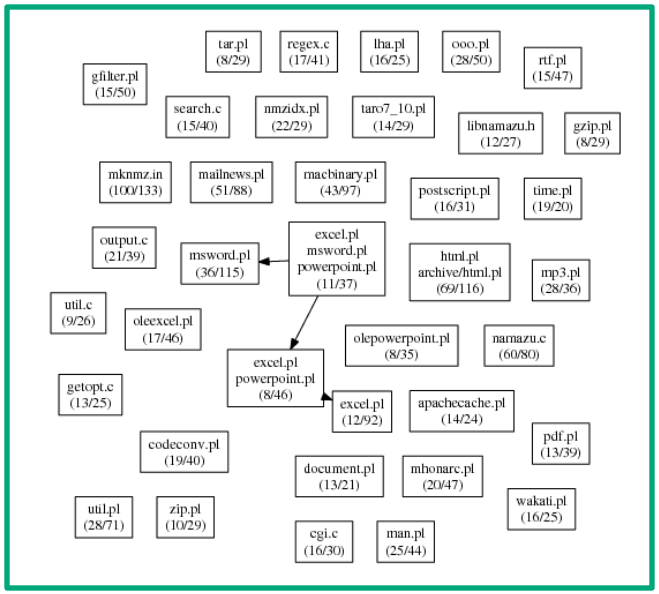
Properties of PRUNIA:

1. The representation of super-sub concept order is persevered
2. No redundant path

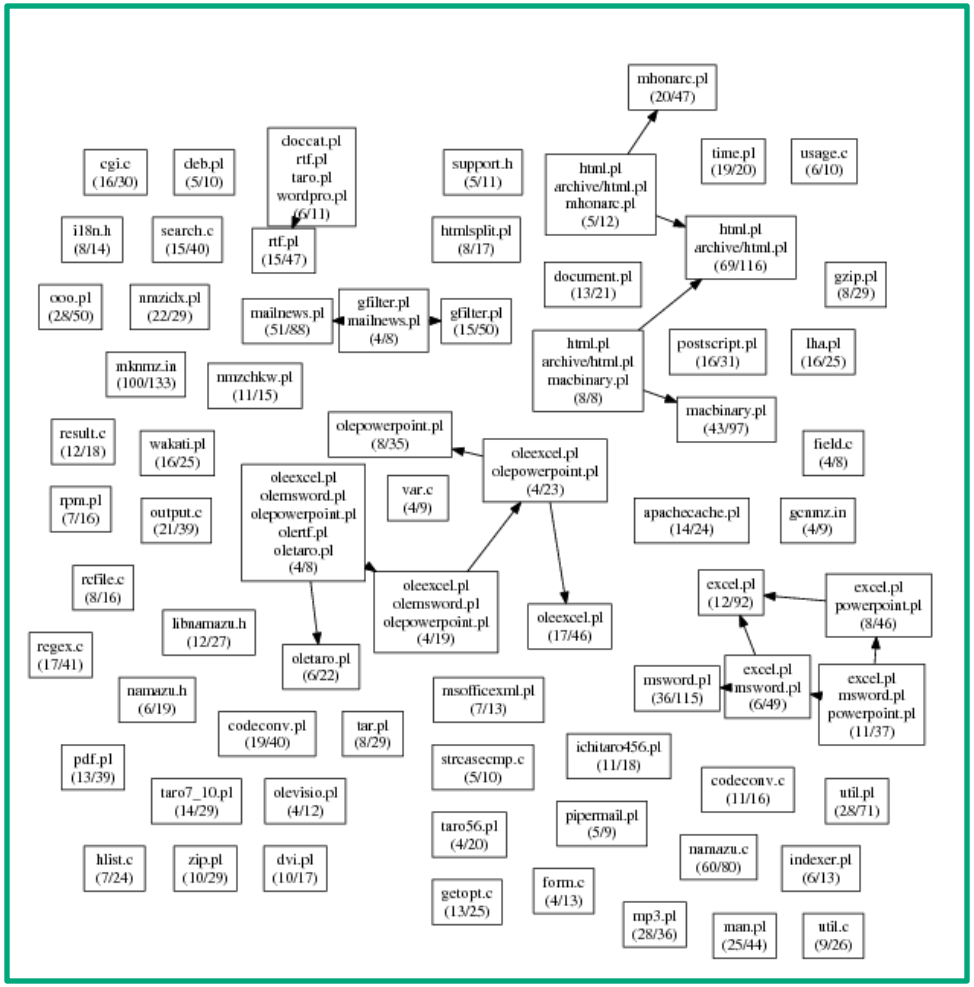
枝刈り結果(Namazuの場合)



(σ=30,τ=15)

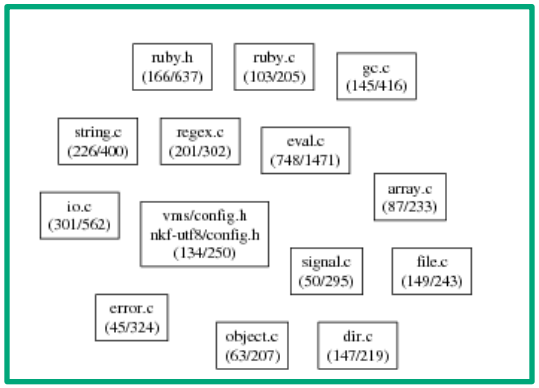


(σ=20,τ=8)

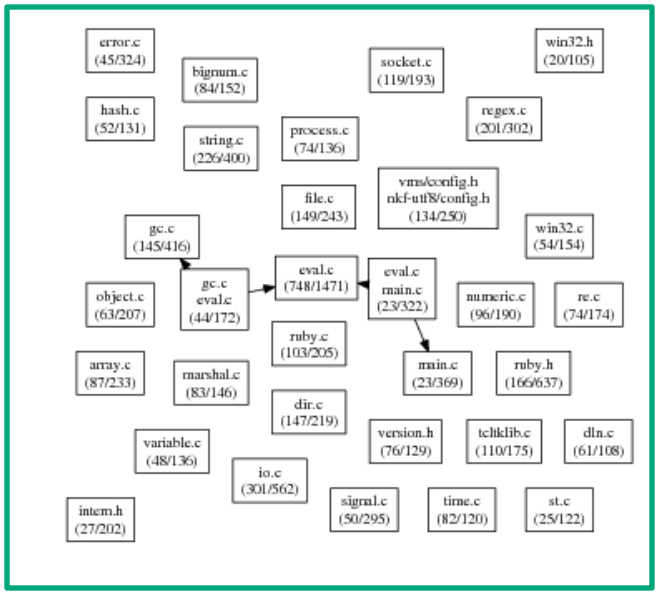


(σ=8,τ=4)

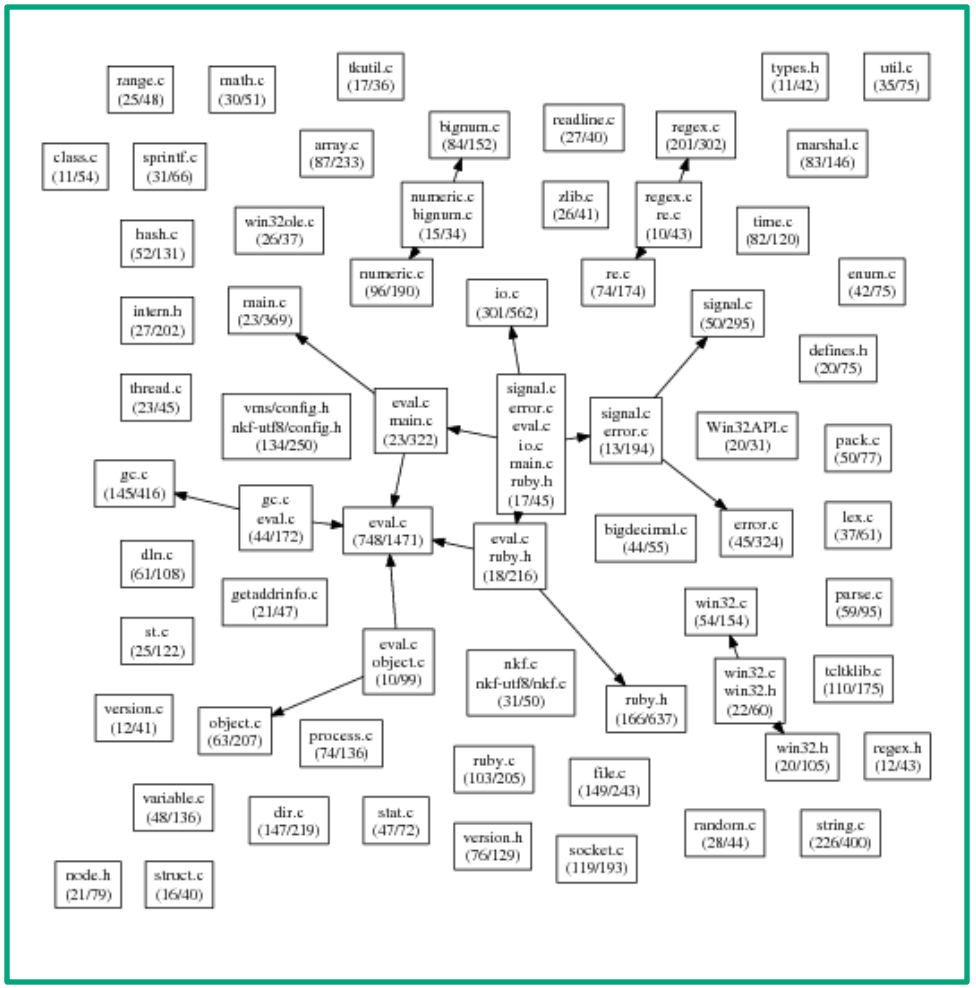
枝刈り結果(Rubyの場合)



(σ=200,τ=40)



(σ=100,τ=20)



(σ=26,τ=10)

(改めて)形式概念とは(1)

- 対象(object)と属性(attribute)の関係をコンテキスト表に表示したときの極大矩形領域

	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}	o_{11}	o_{12}
a_1	●	●	●	●	●	●	●	●	●	●		
a_2	●	●	●	●			●	●				
a_3	●	●	●		●	●					●	●
a_4	●	●		●	●	●					●	●

- グラフ理論の用語では

  2部グラフの完備部分グラフ

(改めて)形式概念とは(2)

形式概念解析(Formal Concept Analysis)

- 数学者 Rudolf Wille が考案
  - その名の通り，”概念”の形式化(数学化)
- 代数学におけるGalois対応を参考にしたといわれる
  - Galois対応：５次以上の代数方程式の解の公式が四則演算とn乗根だけを使って表現できないことを示す際に用いる概念
  - しかし形式概念では数の演算を一切使わない
  - それゆえ，様々な数学理論がこの枠に入る

    数理論理学，形式言語理論,…

数学者向けの説明

- A binary relation $R \subseteq O \times A$
- $f : O \to A$ and $g : A \to O$

$$f(S) = \{\ a \in A \mid (o, a) \in R \text{ for some } o \in S\ \}$$

$$g(T) = \{\ o \in O \mid (o, a) \in R \text{ for some } o \in T\ \}$$

- Then both $h(S) = g(f(S))$ and $k(T) = f(g(T))$ are closure operators.
- A pair of a closed set S in O and $f(S)$ (T in A and $g(T)$) is called a formal concept.

データマイニング，複雑系

- データマイニング(バスケット分析)において形式概念は飽和アイテム集合とよばれている
  - 大量に生成される頻出アイテム集合の同値類
- 複雑系においてはChuシステムとよばれている

形式概念解析の応用例(2)

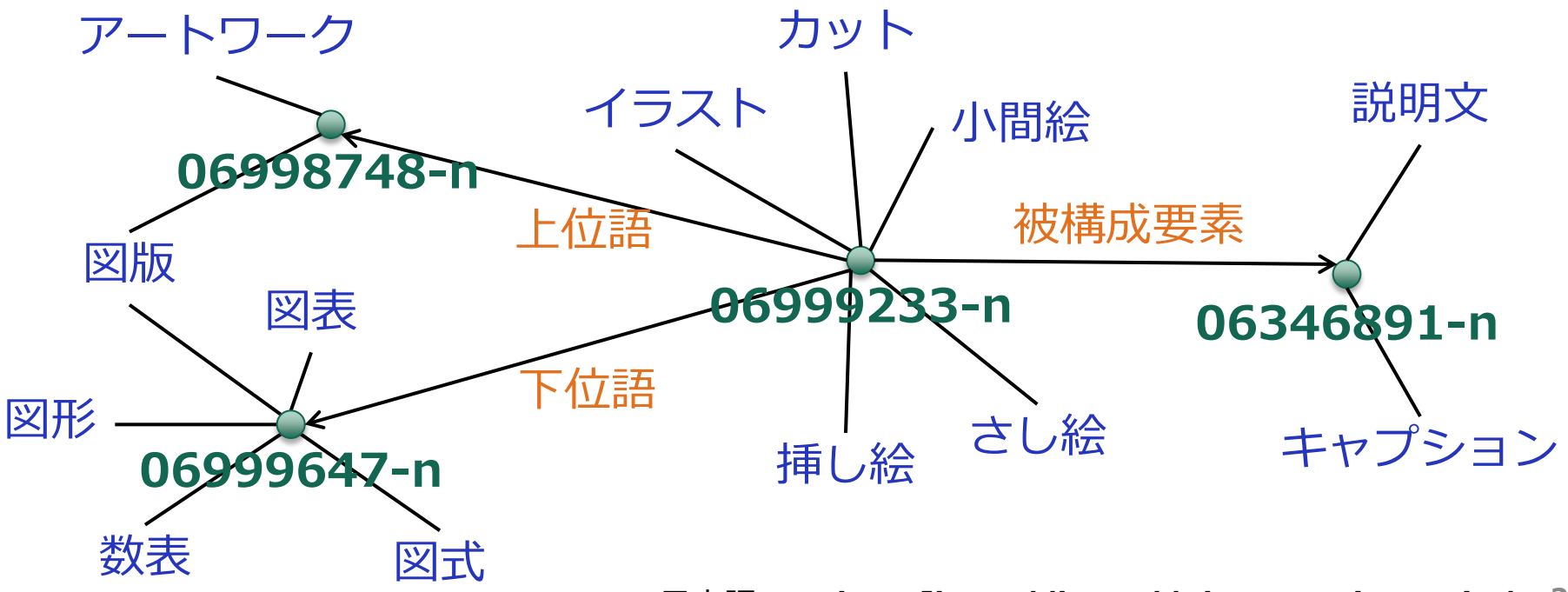
形式概念解析を用いたシソーラス拡張

池田 et al. : クラス分類問題における形式概念解析を用いた近傍決定手法, 2013

- シソーラス拡張:
  既存のシソーラスに登録されていない新たな単語をどの意味に登録するか？
  - 一つの単語が複数の意味を持つ可能性がある

シソーラス

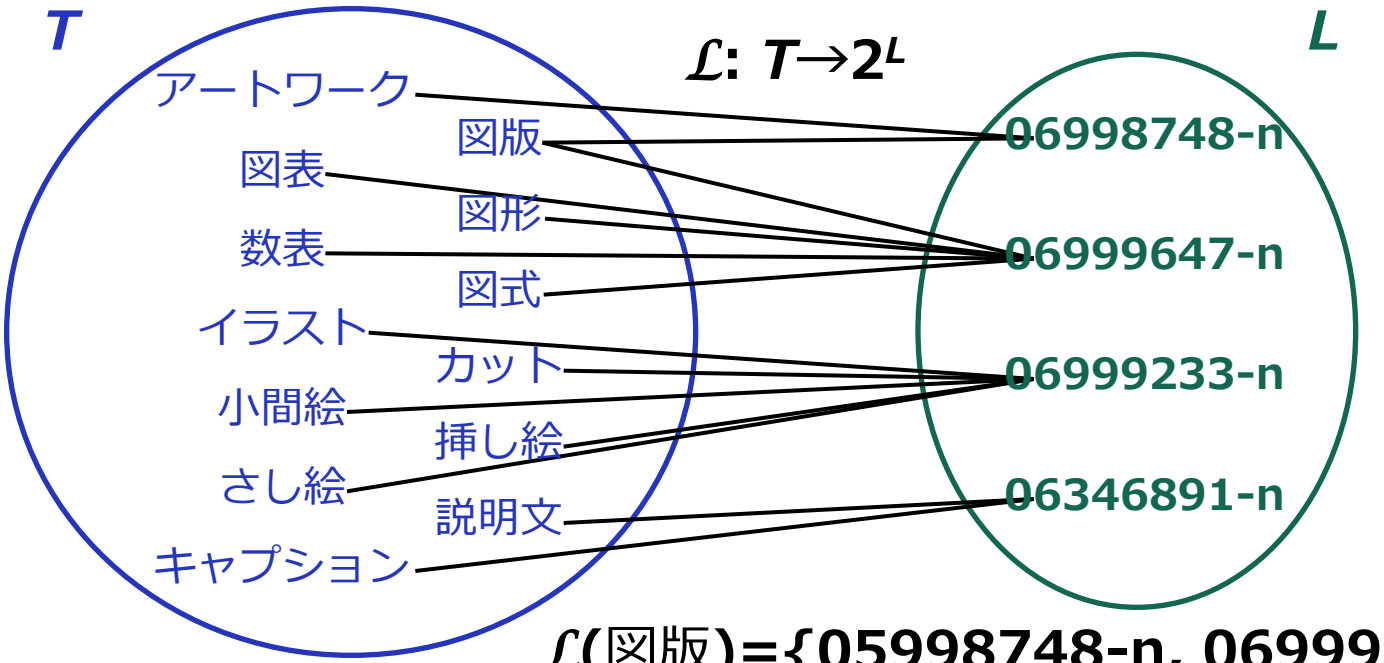
- 語の意味的な辞書
  - 語は意味ラベルに関連付けされている
  - 意味ラベル間には意味関係が与えられる
- 自然言語文の意味解析に不可欠
- 登録語数が限られているため，拡張が必要



日本語WordNet [http://http://nlpwww.nict.go.jp/wn-ja

シソーラス(T, L, $\mathcal{L}$)

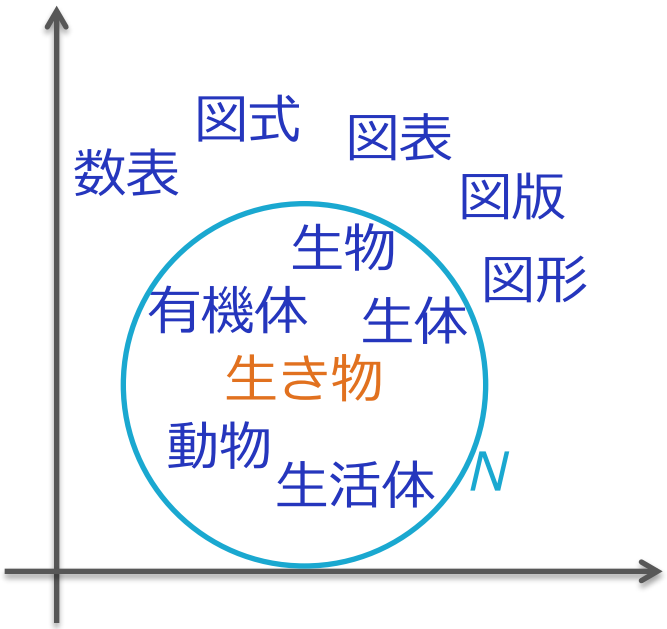
- 語の意味的な辞書
  - 語は意味ラベルに関連付けされ，意味ラベル間には意味関係が与えられる
    - マルチラベルデータ
- 自然言語文の意味解析に不可欠
- 登録語数に限界があるため，拡張が必要



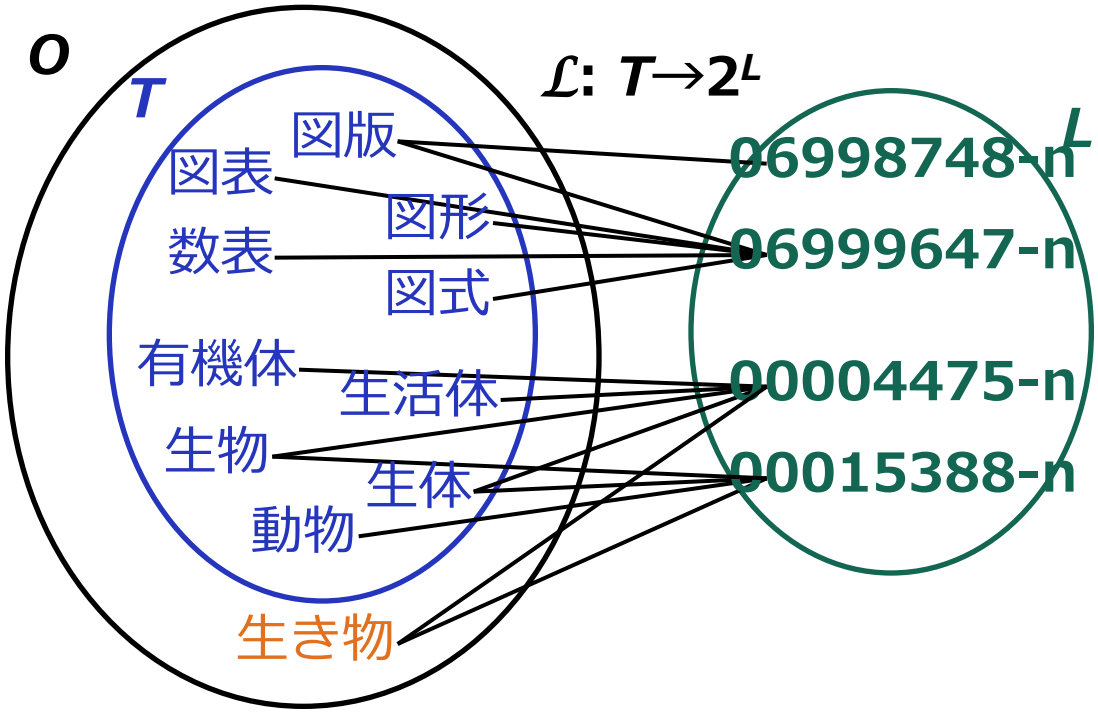
$\mathcal{L}$(図版)={05998748-n, 06999647}

シソーラス拡張問題

- シソーラス(***T, L, ℒ***)の未知語***u*∈*T***に対して***ℒ(u)***を決定
  1. 語の特徴空間上で近傍***N*⊆*T***を決定
  2. 近傍***N***に含まれる既知語***t*∈*T***のラベルから***ℒ(u)***を推定
- シソーラスを教師データとするマルチラベル分類問題



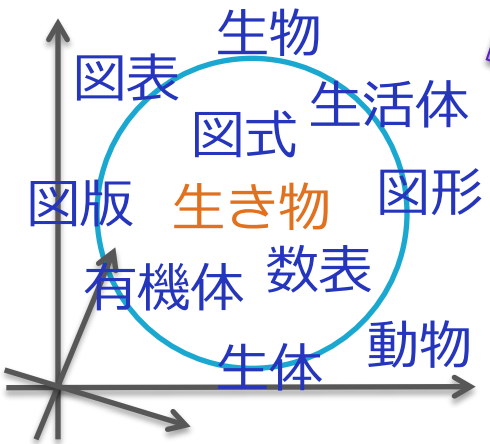
特徴空間

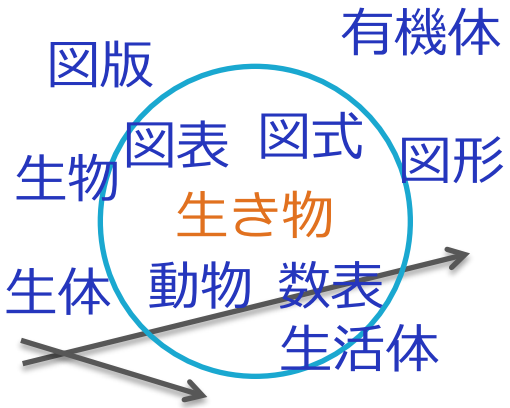


シソーラス拡張に用いる語の特徴

- 自然言語処理においては，自然言語文から構文解析器などの利用により得られる情報が，語の特徴として用いられる
  - 容易に多種多様(品詞，係り受け関係，格構造，***N***-グラム，頻度，．．．)かつ大量のデータ(***O***, ***A***, $\mathcal{A}$)を収集可能
  - 精度向上や計算量削減のため，特徴選択が必要

語の集合***O***	特徴集合***A***
生き物	$\mathcal{A}$(生き物)
生物	$\mathcal{A}$(生物)
図式	$\mathcal{A}$(図式)
：	：

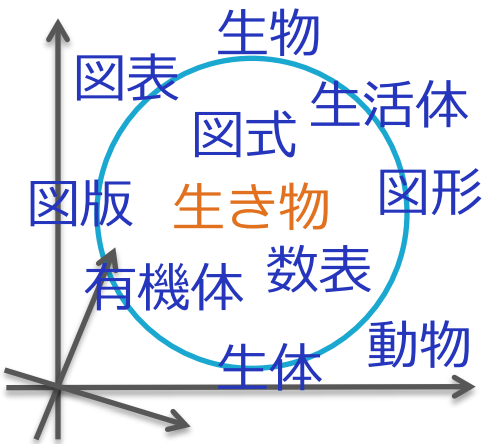


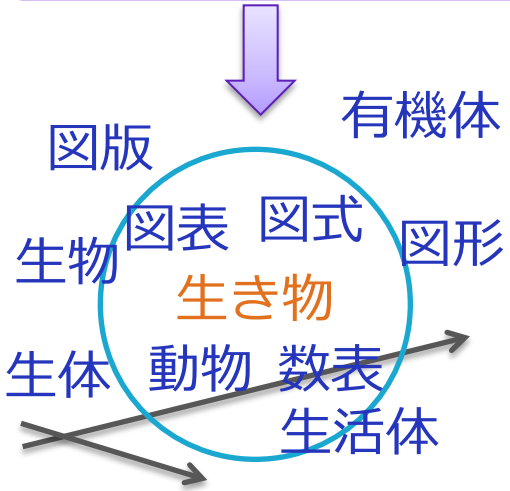


シソーラス拡張に用いる語の特徴

- 自然言語処理においては，自然言語文から構文解析器などの利用により得られる情報が，語の特徴として用いられる
  - 容易に多種多様(品詞，係り受け関係，格構造，***N***-グラム，頻度，．．．)かつ大量のデータ(***O, A,*** $\mathcal{A}$)を収集可能
  - 精度向上や計算量削減のため，特徴選択が必要

語の集合***O***	特徴集合***A'***
生き物	$\mathcal{A}'$(生き物)
生物	$\mathcal{A}'$(生物)
図式	$\mathcal{A}'$(図式)
：	：

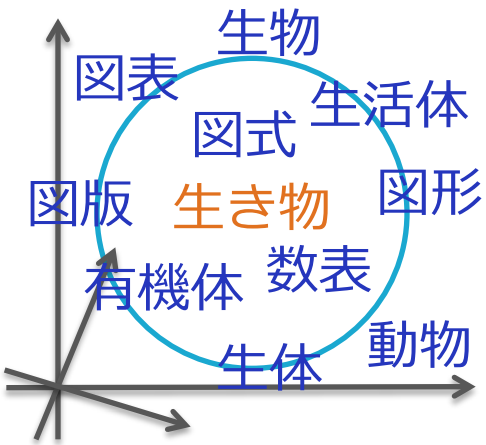


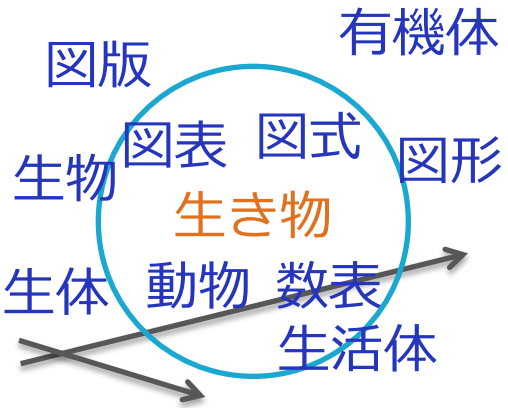


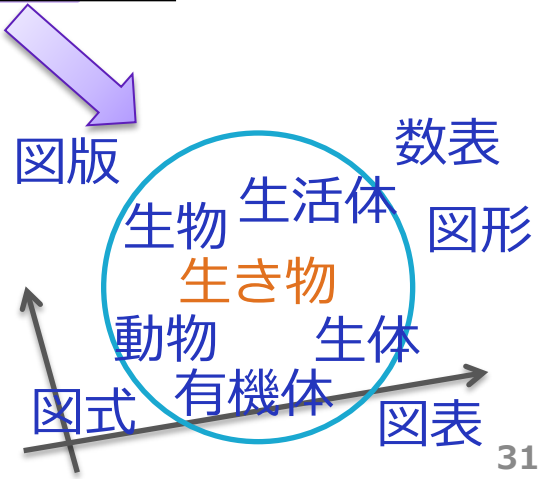
シソーラス拡張に用いる語の特徴

- 自然言語処理においては，自然言語文から構文解析器などの利用により得られる情報が，語の特徴として用いられる
  - 容易に多種多様(品詞，係り受け関係，格構造，***N***-グラム，頻度，．．．)かつ大量のデータ(***O, A, $\mathcal{A}$***)を収集可能
  - 精度向上や計算量削減のため，特徴選択が必要

語の集合***O***	特徴集合***A″***
生き物	$\mathcal{A}$″(生き物)
生物	$\mathcal{A}$″(生物)
図式	$\mathcal{A}$″(図式)
：	：

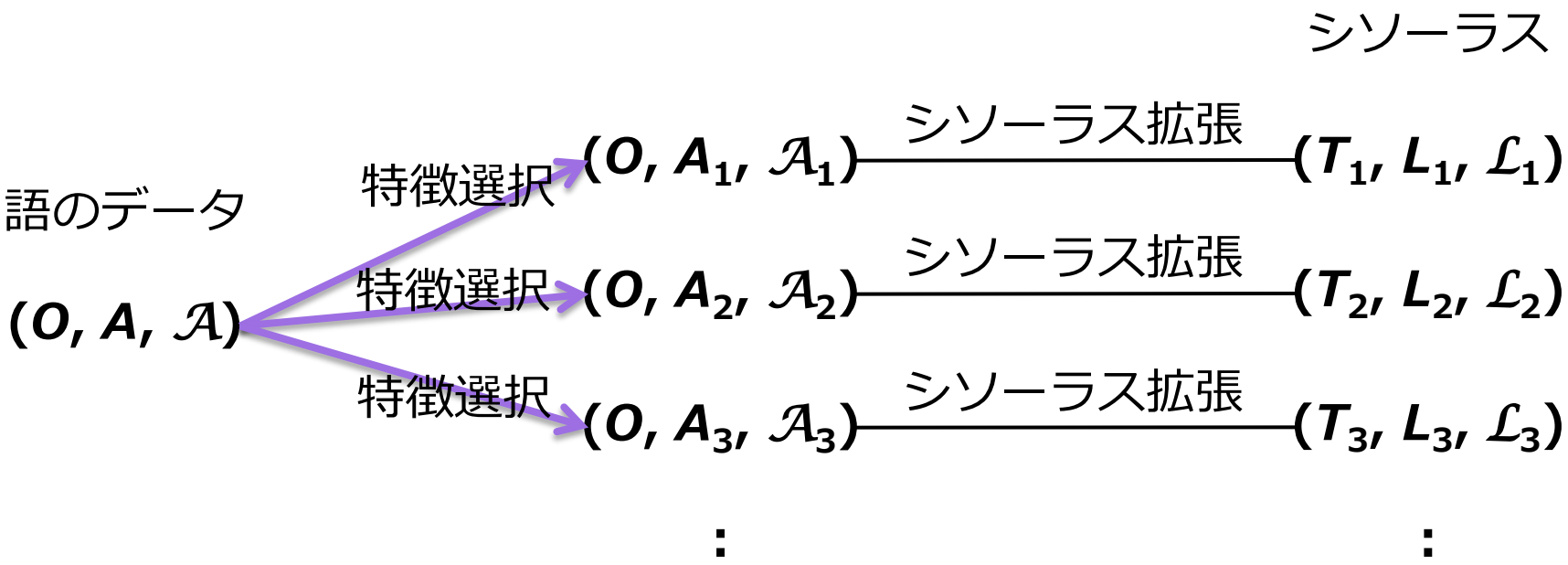






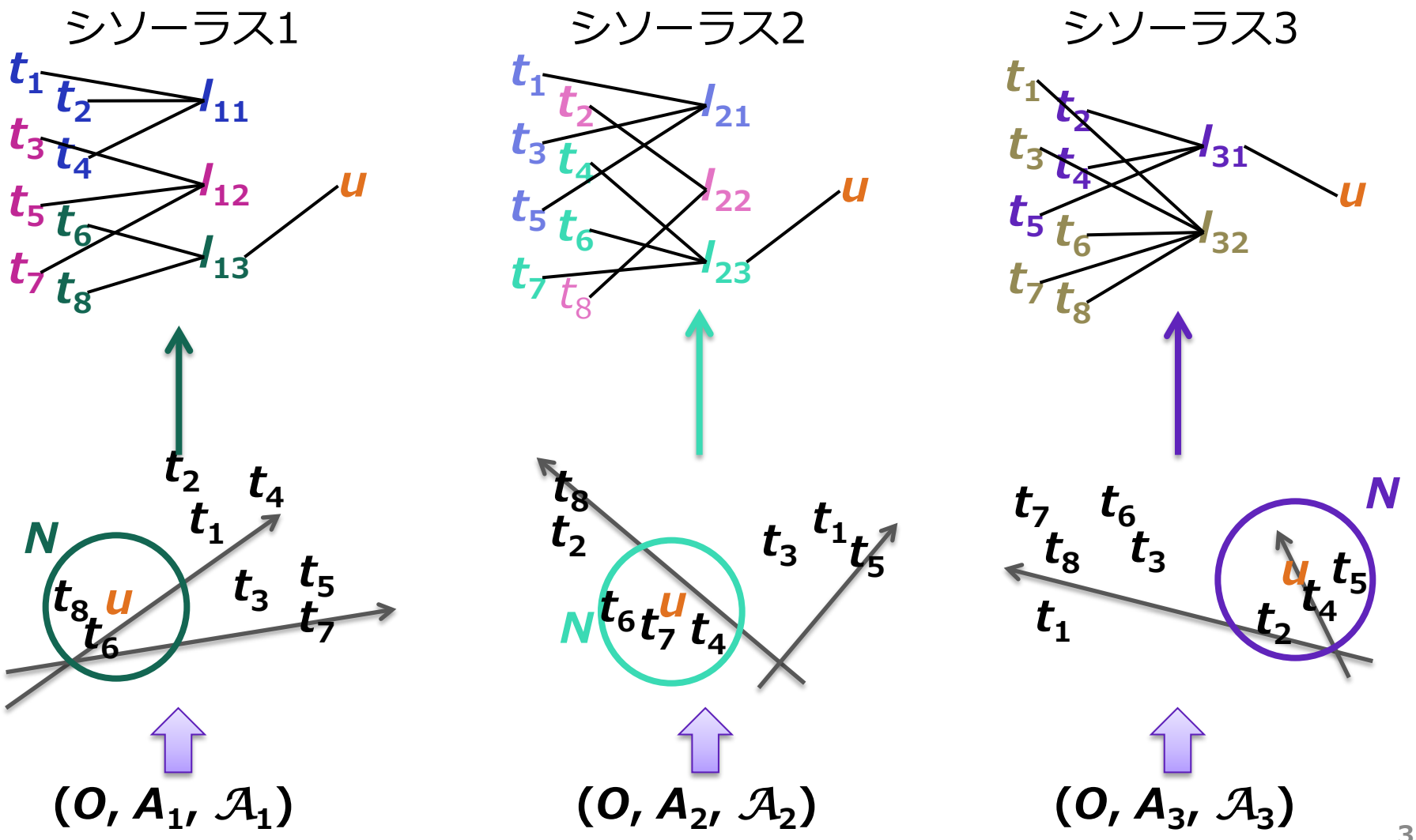
複数のシソーラス拡張問題

- 目的・用途に応じて多種多様なシソーラスが公開されている
  - シソーラス$(T_1, L_1, \mathcal{L}_1)$, $(T_2, L_2, \mathcal{L}_2)$, …が存在し，$T_i \neq T_j$, $L_i \neq L_j$, $\mathcal{L}_i \neq \mathcal{L}_j$
- データ$(O, A, \mathcal{A})$に対して，複数のシソーラス$(T_1, L_1, \mathcal{L}_1)$, $(T_2, L_2, \mathcal{L}_2)$, …を拡張する必要がある
  - シソーラスそれぞれに対して複数の特徴選択が必要
- 拡張すべきシソーラスの数が増えれば，より多くの時間が必要



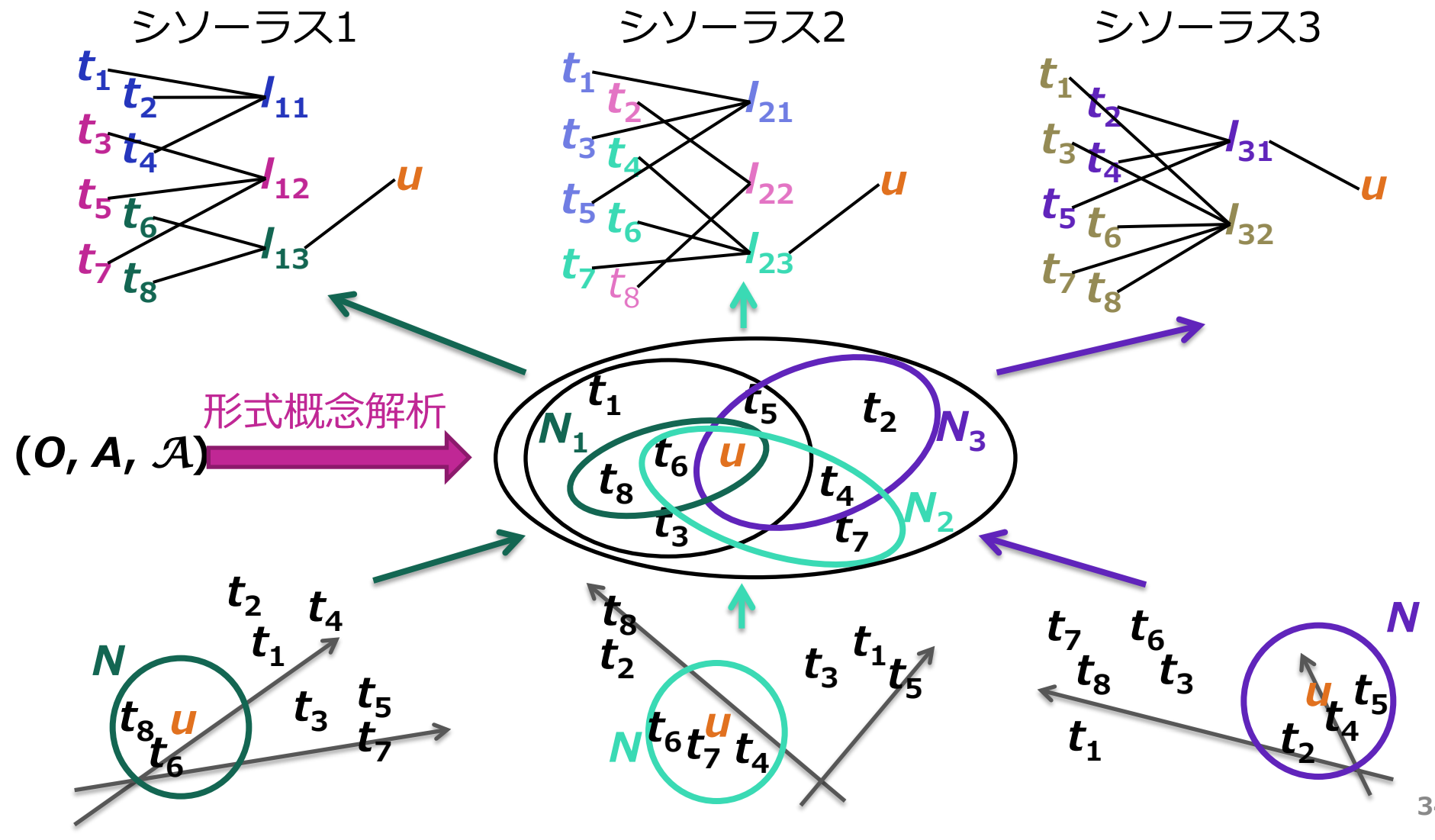
アイデア

1. データから，あらかじめ語の近傍の候補を複数用意
2. 拡張するシソーラスについて，未知語の近傍を候補から選択



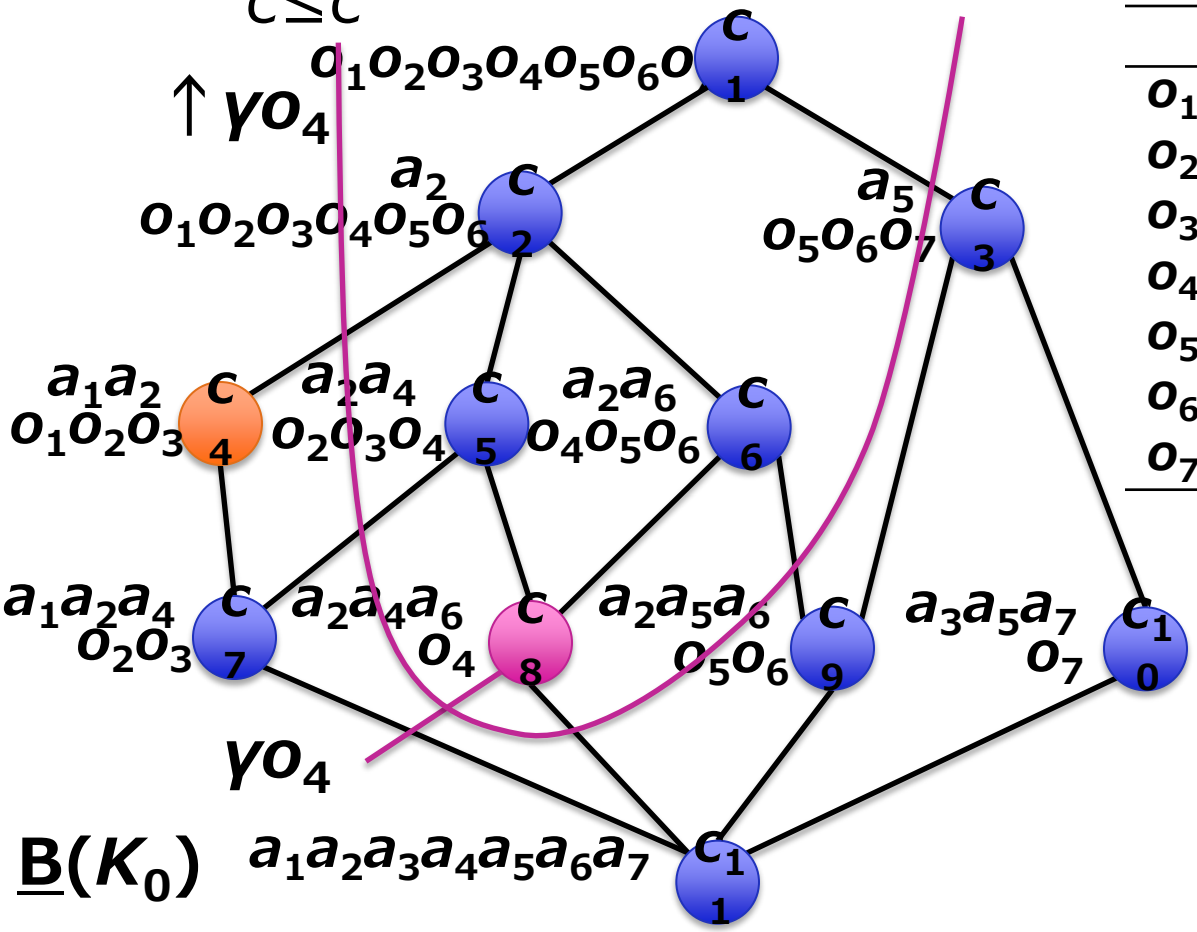
アイデア

1. データから，あらかじめ語の近傍の候補を複数用意
2. 拡張するシソーラスについて，未知語の近傍を候補から選択



概念束

- 形式文脈$K=(O, A, I)$の概念束$\underline{B}(K)$
  - γo: $o \in O$のオブジェクト概念 $(\{o\}^{II}, \{o\}^{I})$
  - $\uparrow c$: $c \in \underline{B}(K)$の上方集合$\{c' \in \underline{B}(K) \mid c \leq c'\}$
    - 形式概念$c, c' \in \underline{B}(K)$について，$\mathrm{Ex}(c) \leq \mathrm{Ex}(c')$ならば $c \leq c'$



	a_1	a_2	a_3	a_4	a_5	a_6	a_7
o_1	×	×					
o_2	×	×		×			
o_3	×	×		×			
o_4		×		×		×	
o_5		×			×	×	
o_6		×			×	×	
o_7			×		×		×

$K_0=(O_0, A_0, I_0)$

形式概念解析を用いたシソーラス拡張

1. データから，あらかじめ語の近傍の候補を複数用意
   - データから求められる形式概念の外延を近傍の候補とする
2. 拡張するシソーラス$(T, L, \mathcal{L})$について，未知語uの近傍Nを候補から選択
   - 形式概念cにスコア$\sigma(c)$を与え，最大のスコアを持ち，かつ外延の要素数が最大である形式概念の外延に含まれる既知語の集合を近傍とする

$$\sigma(c)=\begin{cases} 0 & \text{if } |\mathrm{Ex}(c)\cap T|=0, \\ 1 & \text{if } |\mathrm{Ex}(c)\cap T|=1, \text{ and} \\ \dfrac{\sum_{i=1}^{|\mathrm{Ex}(c)\cap T|}\sum_{j=i+1}^{|\mathrm{Ex}(c)\cap T|}\mathrm{sim}(o_i, o_j)}{\binom{|\mathrm{Ex}(c)\cap T|}{2}} & \text{otherwise,} \end{cases}$$

where

$$\mathrm{sim}(o_i, o_j)=\frac{|\mathcal{L}(o_i)\cap\mathcal{L}(o_j)|}{|\mathcal{L}(o_i)\cup\mathcal{L}(o_j)|}$$

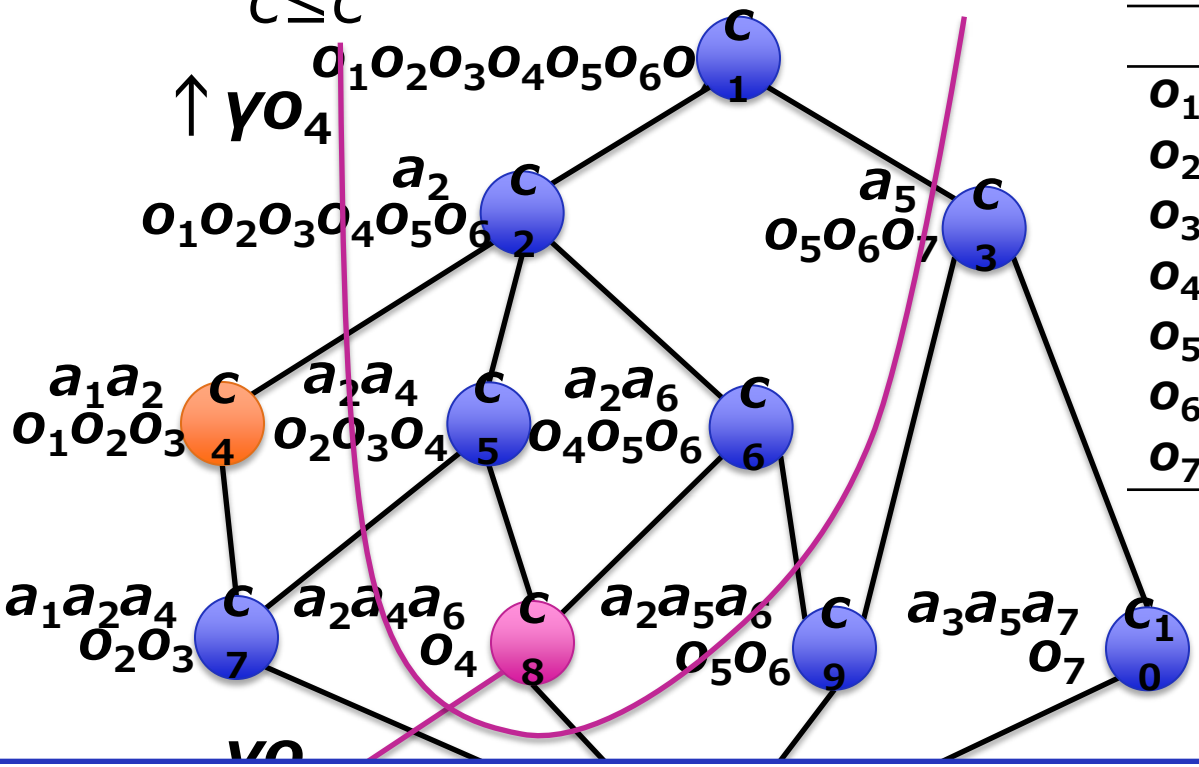
類似度の平均

既知語間の類似度
(Jaccard Index)

3. 未知語uのラベル$\mathcal{L}(u)$を近傍のラベルを基に決定する
   - $\mathcal{L}(u)=\cup_{o\in N}\mathcal{L}(o)$

概念束

- 形式文脈$K=(O, A, I)$の概念束$\underline{B}(K)$
  - γo: $o \in O$のオブジェクト概念 $(\{o\}^{II}, \{o\}^{I})$
  - $\uparrow c$: $c \in \underline{B}(K)$の上方集合$\{c' \in \underline{B}(K) \mid c \leq c'\}$
    - 形式概念$c, c' \in \underline{B}(K)$について，$\mathrm{Ex}(c) \subseteq \mathrm{Ex}(c')$ならば $c \leq c'$

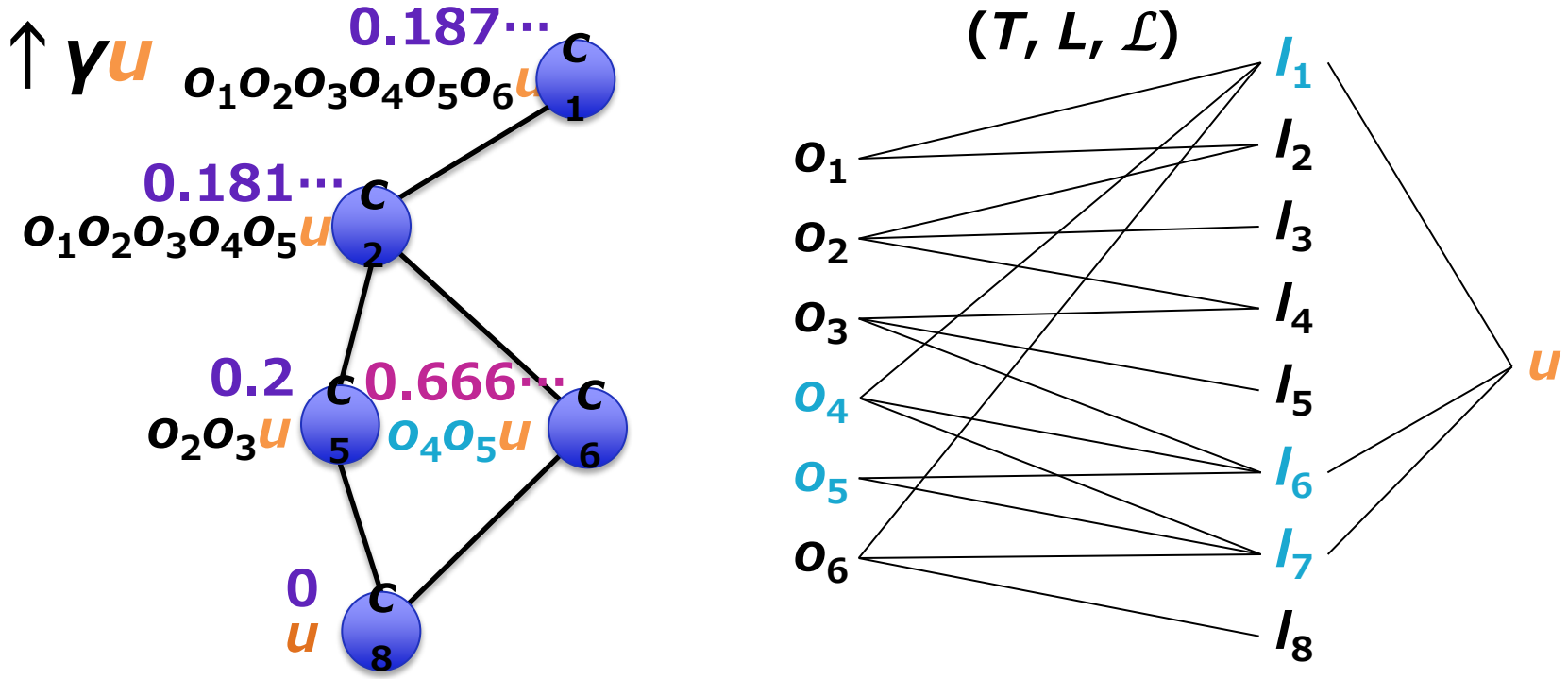


	a_1	a_2	a_3	a_4	a_5	a_6	a_7
o_1	×	×					
o_2	×	×		×			
o_3	×	×		×			
o_4		×		×		×	
o_5		×			×	×	
o_6		×			×	×	
o_7			×		×		×

$K_0=(O_0, A_0, I_0)$

- シソーラス拡張において，未知語が含まれる外延をそれぞれ未知語の近傍の候補とする

形式概念解析を用いたシソーラス拡張



$\sigma(c_2)=(\mathrm{sim}(o_1, o_2)+\mathrm{sim}(o_1, o_3)+\mathrm{sim}(o_1, o_4)+\mathrm{sim}(o_1, o_5)+\mathrm{sim}(o_2, o_3)+\mathrm{sim}(o_2, o_4)+\mathrm{sim}(o_2, o_5)+\mathrm{sim}(o_3, o_4)+\mathrm{sim}(o_3, o_5)+\mathrm{sim}(o_4, o_5))/10$

$=(0.25+0+0.25+0+0.2+0+0+0.2+0.25+0.666\cdots)/10$

$=0.181\cdots$

- スコアを用いた近傍の決定は暗黙的な特徴選択といえる

実験

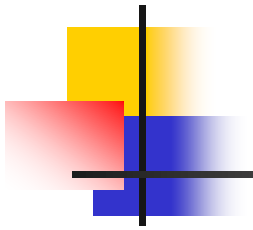
- 実験用データ
  - シソーラス(***T, L, ℒ***)
    - 日本語**WordNet (JWN)**，分類語意表 **(BGH)**
  - 語のデータ(***O, A, 𝒜***)
    - 京都大学格フレームコーパス **(KCF)**
    - 日本語***N*-グラム (*N*=4) (J4g)**
    - **KCF・J4g**混合データ

- 評価方法
  - シソーラスとデータの組み合わせ**(6**種**)**について交差検定**(10-fold)**を行ない適合度**(precision)**と再現度**(recall)**を計測
  - ***k*近傍法(*k*=1, 5, 10)**と比較

結果

- 全ての実験において***k***近傍法より良い結果が得られた

		JWN		BGH	
		precision	recall	precision	recall
KCF	提案手法	0.039	0.274	0.164	0.553
	1-NN	0.026	0.024	0.103	0.103
	5-NN	0.007	0.036	0.031	0.150
	10-NN	0.004	0.038	0.016	0.169
J4g	提案手法	0.007	0.079	0.028	0.248
	1-NN	0.007	0.007	0.027	0.027
	5-NN	0.002	0.013	0.014	0.070
	10-NN	0.002	0.018	0.010	0.100
混合	提案手法	0.030	0.072	0.132	0.250
	1-NN	0.009	0.009	0.039	0.039
	5-NN	0.004	0.018	0.017	0.085
	10-NN	0.002	0.024	0.011	0.116



まとめ

- 形式概念解析は二元論
  - 1個の情報源だけでなく，2個の情報源の関係をみる
- 形式概念解析の利用者の先輩が存在
  - 数学，複雑系，自然言語解析
- 形式概念解析は対象間の演算を仮定しない
  - 演算を導入することでより深い解析が期待される
- 形式概念解析は情報圧縮の一種
  - プライバシー保護への展望