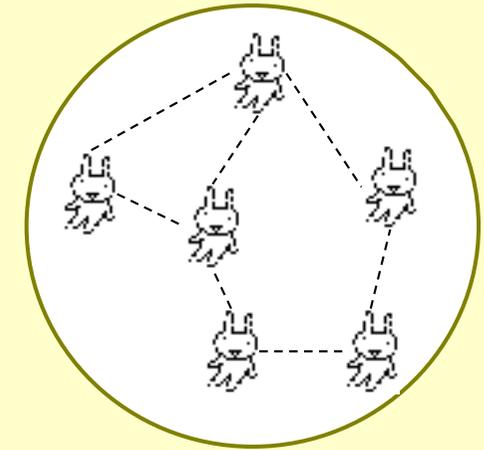


# 構造データの機械学習

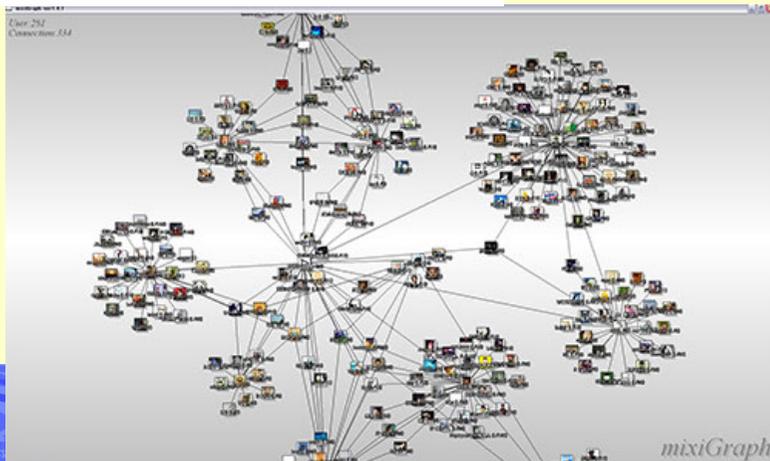
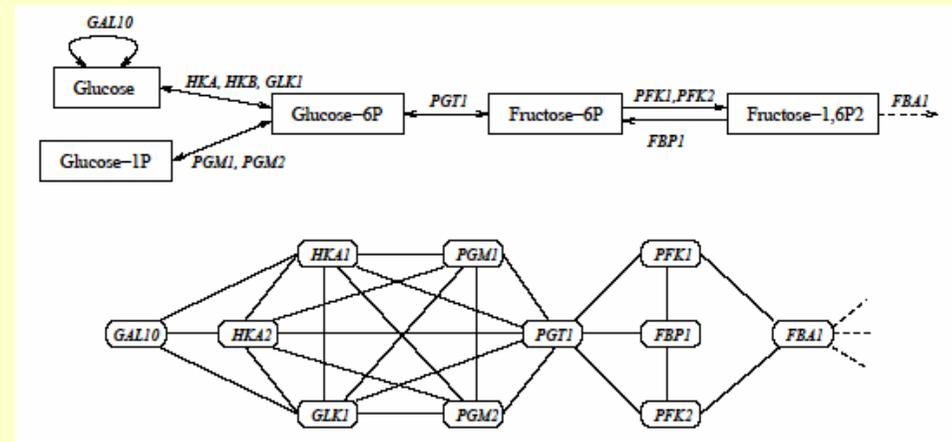
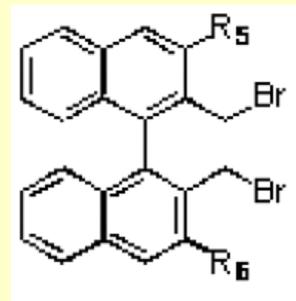
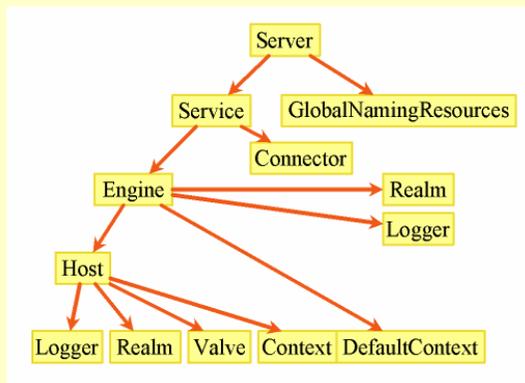


鹿島 久嗣

IBM / 東京基礎研究所

# 今日は、構造をもったデータを扱う機械学習法についてお話しします

- 構造をもったデータとは、たとえば、
  - 配列データ： DNA、タンパク質、自然言語、イベント列、時系列
  - 木構造データ： HTML/XML、RNA構造、構文解析木、系統樹、ディレクトリ
  - グラフ構造データ： 化合物、WWW、社会ネットワーク、生体ネットワーク

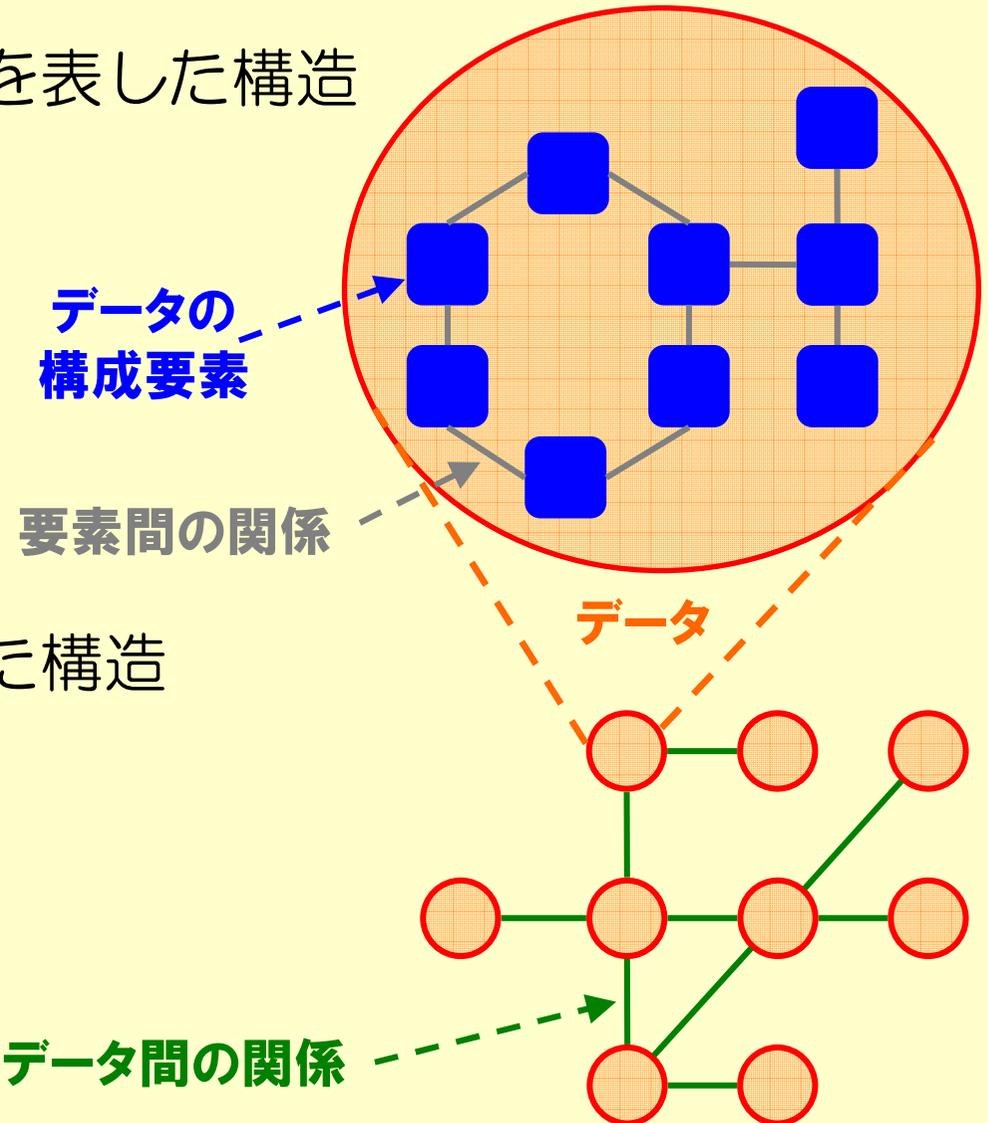


```
MSGVYKVSGIQSILQKDVTSEGETAILISLGLMTKEEKVPVPAKMAMVASA 50  
KANSIIFVSEDGSLSFEPKETGETSKPGEKKEEKKVEVGKFPFSAAKV 100  
KELIEGKSLTLDQDKIQKVL EEYVKNLPRTAETYKPKIEIKCFKGVDFS 150  
ISSLLSSGTKILDAILYSTYKDSA EHNIFDVKVLSPDFIDSKLLVNNIE 200  
TGNRAIKAAFCLVYNQGLPSKTSEERPLSKFVRETIFREKDLKANELCE 250  
YLSSADPSLFP SQVFLKISLENLPT EVSSRCKMSIAGNKAMRYALLAQKF 300  
DKDEI PVPTEVNPTTSSEYMCKKEKIEKAKKIVDVLCSLASDFQAQVKMH 350  
PLSPERSRKNFTLQLTSAIVTSLSYKGR LDMRKAIEEKKIEAFKRDENI 400  
FGRLNALGQPTFPVLTNADADFSELSVEAVK TAYGKK 437
```

# 「構造」には内部構造と外部構造の2種類があります

- 内部構造: データ内の要素の関連を表した構造

- HTML/XML
- DNA
- 化合物



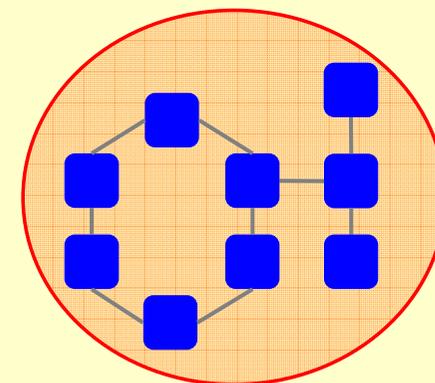
- 外部構造: データ間の関連を表した構造

- Web
- 社会ネットワーク
- 遺伝子/タンパク質ネットワーク

# 前半は内部構造の扱いについて、 後半は外部構造の扱いについて話します

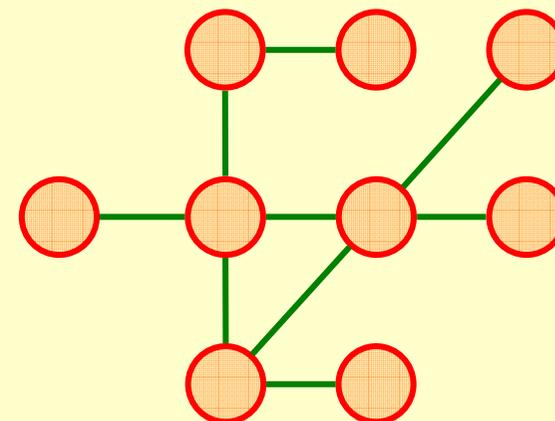
## ■ 内部構造の扱い

- カーネル法に基づく解析手法
  - ラベル付順序木カーネル
  - グラフ・カーネル



## ■ 外部構造の扱い

- 複数タイプのリンク予測



## 教師付き(分類)学習を前提にお話します

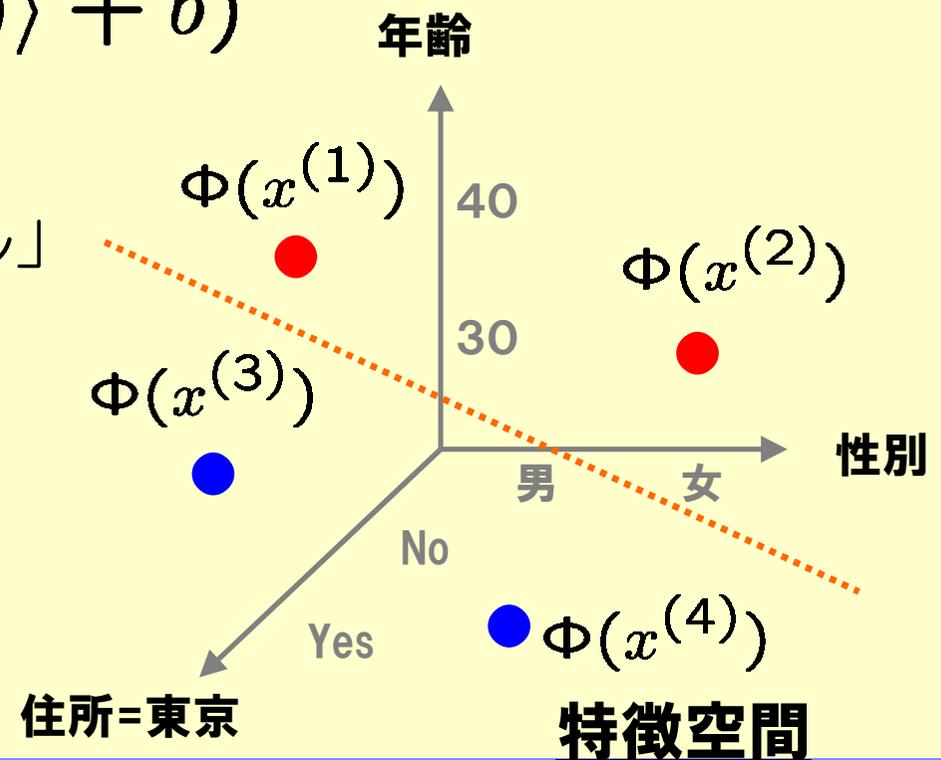
- 教師つき学習は、+1 か -1 の出力ラベルのついた訓練データ  $\{(x^{(1)}, +1), (x^{(2)}, +1), (x^{(3)}, -1), (x^{(4)}, +1), \dots\}$

をもとに、ラベル未知のデータ  $x$  のラベルを予測する予測器

$$h(x) = \text{sign}(\langle \mathbf{w}, \Phi(x) \rangle + b)$$

を学習する

- $\Phi(x)$  は、 $x$  の「特徴ベクトル」
- 予測器は、特徴空間中を分割する超平面とする
  - $\mathbf{w}$  と  $b$  はモデルパラメータ



---

## 内部構造の扱いについて

## 内部構造の扱いについて、これからお話すること

---

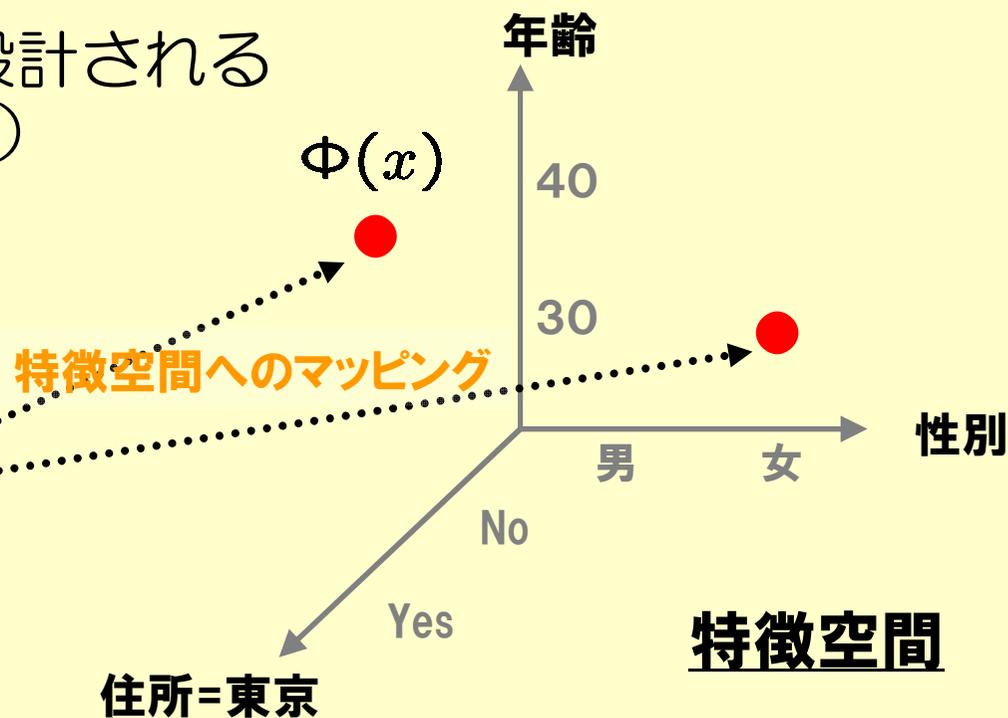
- (内部) 構造データを扱う学習が、なぜ難しいか？
- 便利な枠組み：カーネル法
- 順序木を扱うためのカーネル法
- グラフを扱うためのカーネル法

従来の機械学習手法ではベクトル型のデータを前提としているため、構造をもったデータをうまく扱うことが出来ません

- 従来の機械学習手法では、データ  $x$  が特徴空間中のベクトル  $\Phi(x)$  として表されていることを前提とする
- 配列／木／グラフなどの構造をもったデータでは、一般的な特徴空間の構成が自明でない
  - ドメインによって、独自に設計される  
(例：化合物におけるQSAR)

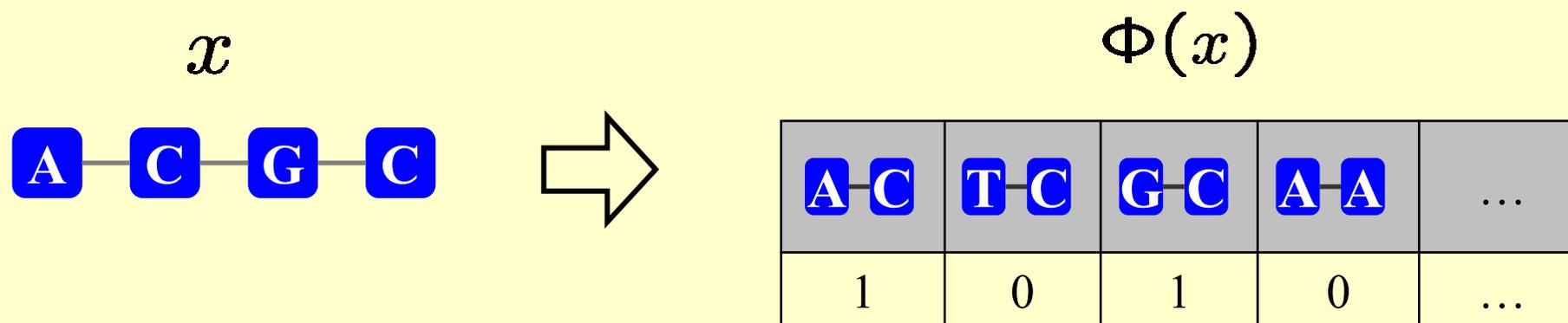
### 従来：ベクトル型のデータ

顧客番号	顧客氏名	年齢	性別	住所	...
0001	〇〇	40代	男性	東京都	...
0002	××	30代	女性	大阪府	...



1つの考えるアプローチは「部分構造」を用いて特徴空間を構成することですが、計算量的な問題を抱えています

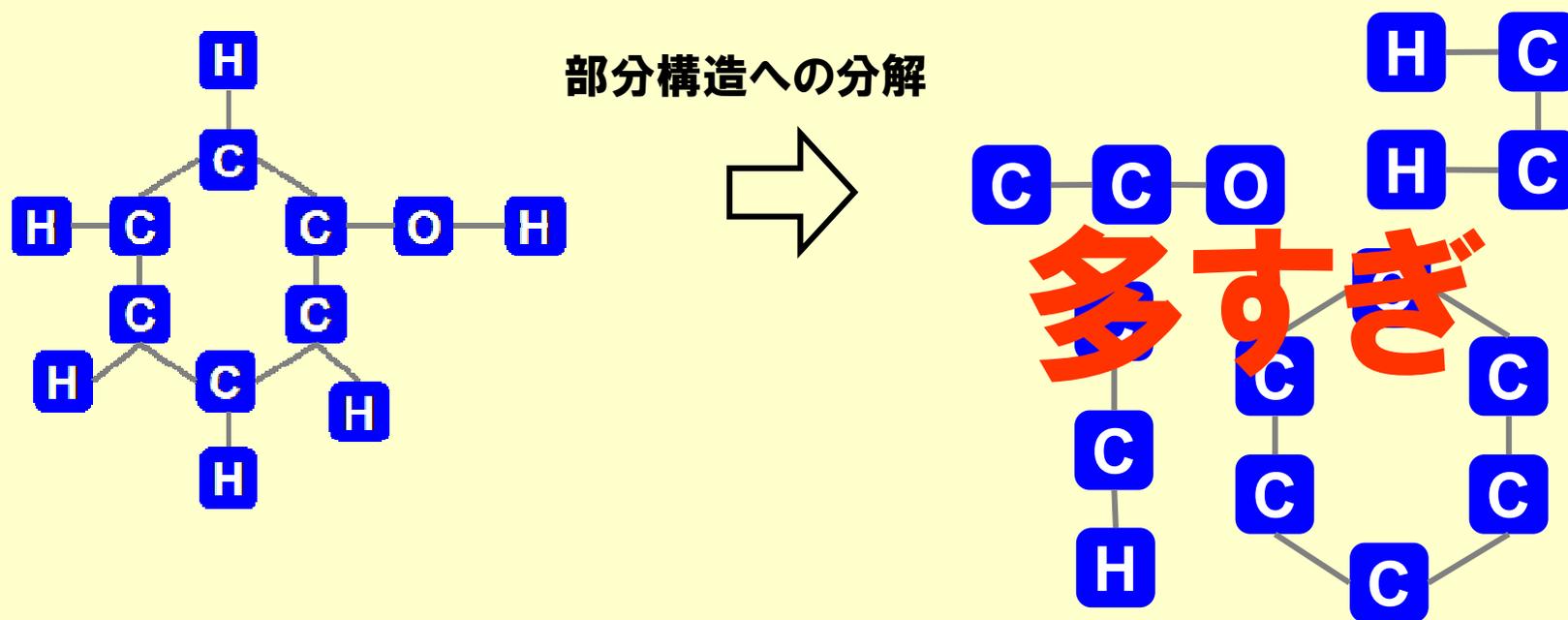
- 「部分構造」が構造の性質を担っていると考える
  - 配列データの性質は、部分配列が担う（＝マルコフモデル）
- 各部分構造の有無や出現回数を用いて、特徴ベクトル表現する
- しかし、すべての部分構造をテーブル要素の候補にするときりがない
  - グラフには、指数個の部分グラフが含まれるので、すべてを数え上げている場所と時間はない（これが本質的な問題）



「カーネル法」は「部分構造多すぎ問題」を、比較的キレイに解決してくれる枠組みです

■ 考えるアプローチ：

- 部分構造のサイズを限定する ← マルコフモデル
- 何らかの条件をつけて限られた重要なものだけを数え上げる？  
← パタンマイニング
- そもそもベクトル表現に持ち込まない？ ← カーネル法



# カーネル法とは、データアクセスが、特徴空間の次元に依存しない学習器のクラスです

- カーネル法はカーネル関数（＝特徴空間における内積）によってのみデータアクセスを行う学習器のクラス

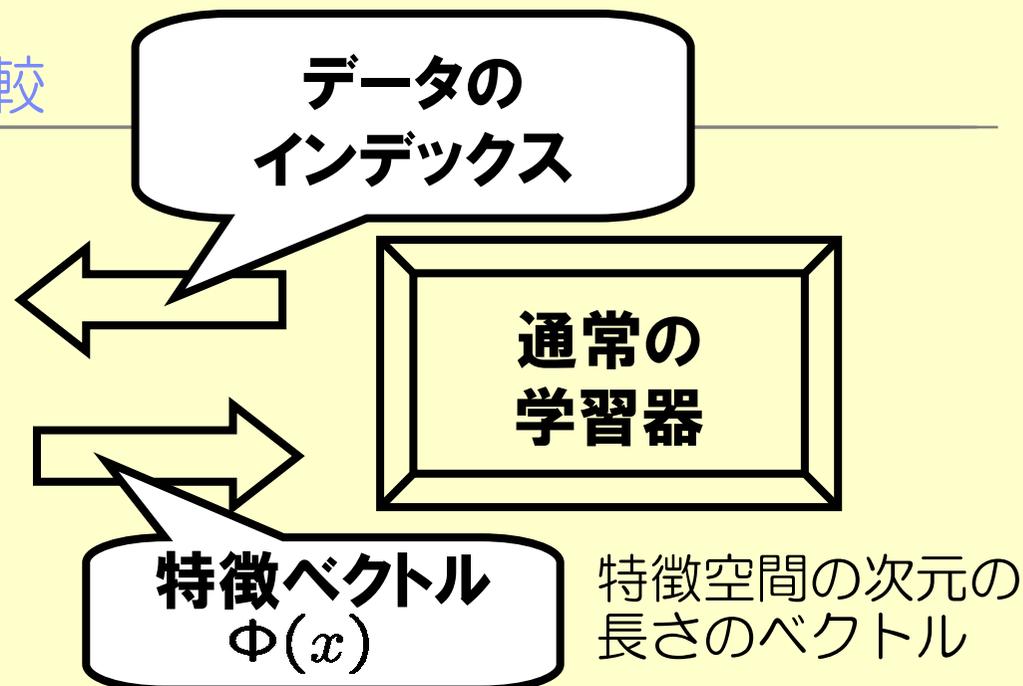
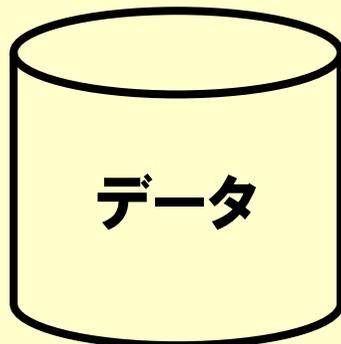
$$h(x) = \text{sign} \left( \sum_{i=1}^N \alpha_i K(x^{(i)}, x) \right) \quad (\text{カーネル予測器：パラメータ } \alpha)$$

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle \quad (\text{カーネル関数：類似度})$$

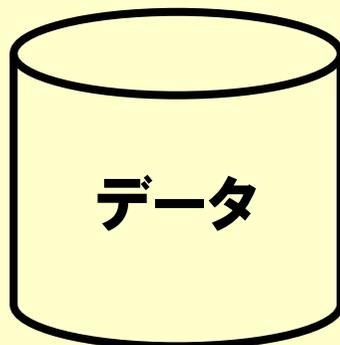
- 観察：特徴ベクトル  $\Phi(x)$  へのアクセスが常にカーネル関数  $K$  を経由している
- ポイント：特徴ベクトル  $\Phi(x)$  が陽に現れないため、特徴ベクトルを陽に構成する必要が無い
  - カーネル予測器からみると、データアクセス部分が次元に依存しない
  - $x$  や  $x'$  が何であれ、適当に類似度  $K(x, x')$  を定義すればそれがカーネル関数として使える、ともいえる
    - ただし、内積になっていることが保証される必要はある

# イメージ： カーネル法と非カーネル法の比較

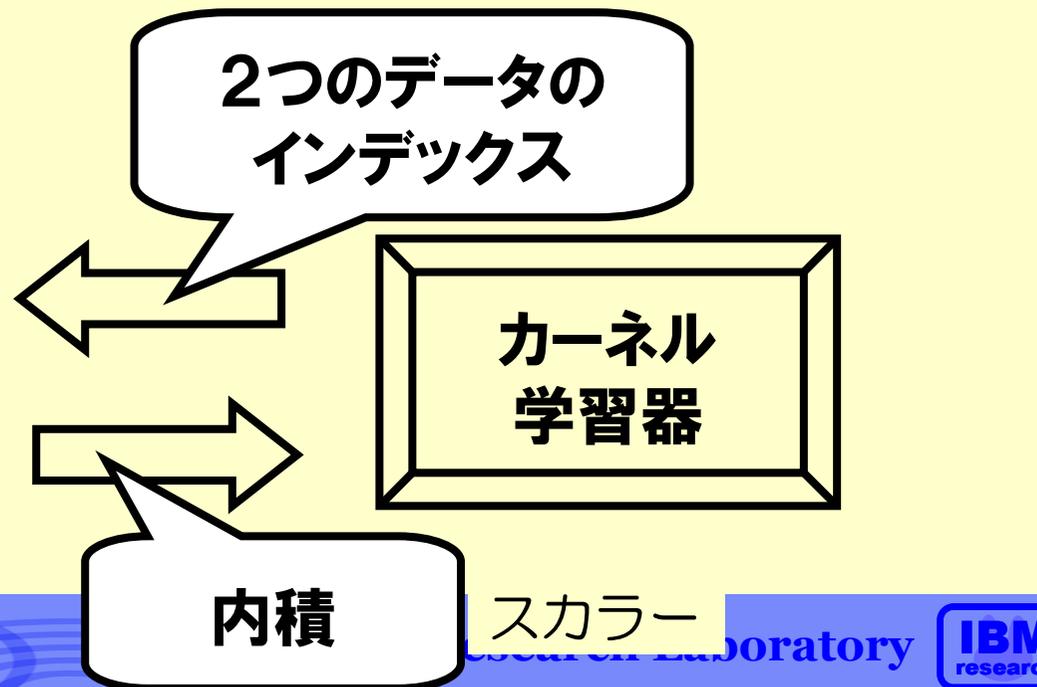
非カーネル法



カーネル法



カーネル関数



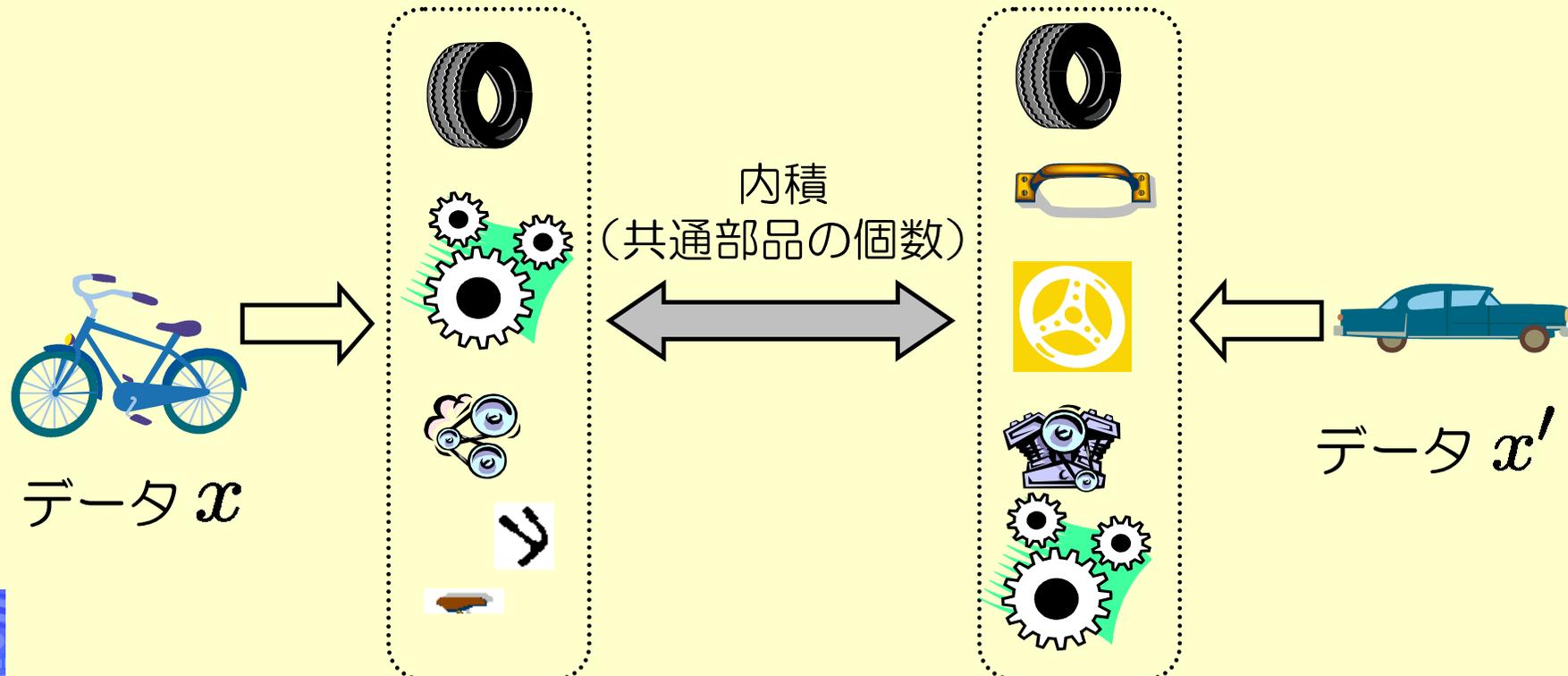
## 結局のところ、ポイントは、カーネル関数の設計になります

---

- カーネル法とは、
  - 学習器
  - カーネル関数 (=2つのデータの類似度)が分離している機械学習法
- カーネル法は、良いカーネル関数を定義できれば、あとはおまかせで動く → ポイントはカーネル関数の設計
- 構造データに対する場合も自然に扱うことができる
  - 配列に対するカーネル関数
  - 木に対するカーネル関数
  - グラフに対するカーネル関数
  - . . .

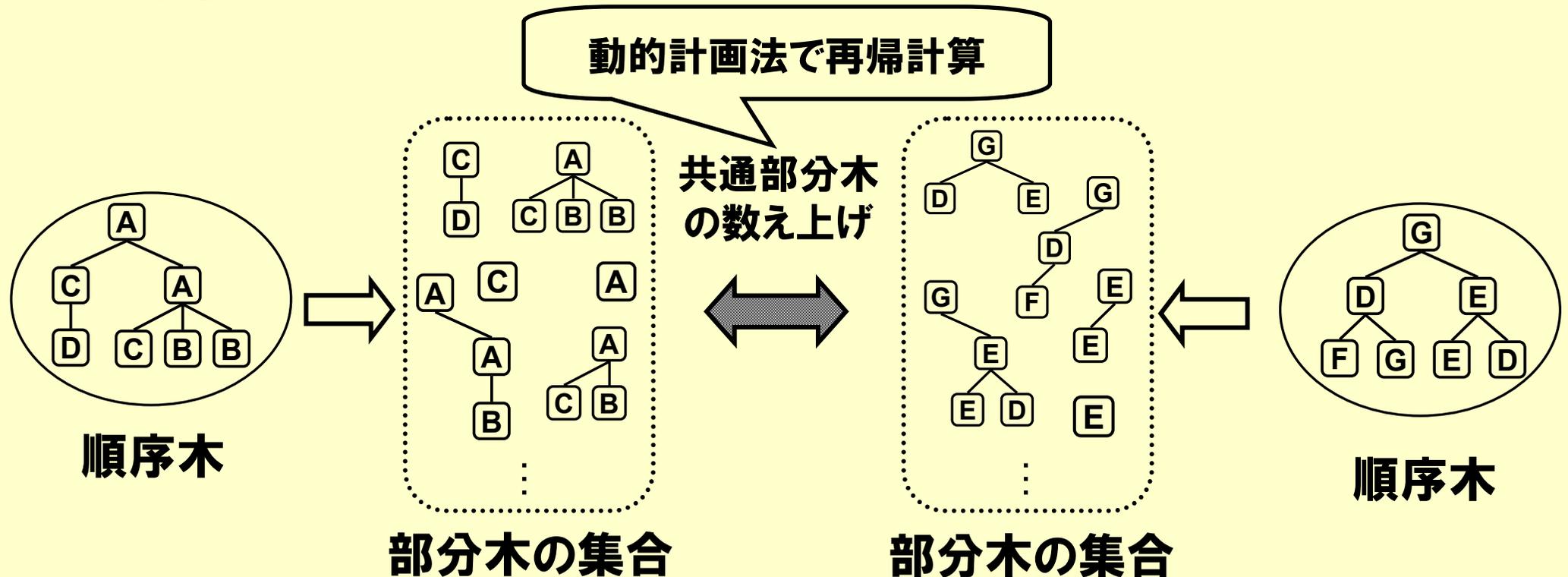
## 私は、順序木とグラフに対するカーネル関数を設計しました

- 2つの構造データのカーネル関数（=部分構造によって定義された2つの特徴ベクトルの内積）を、特徴ベクトルを明示的に作らずに計算する
  - 対象とする構造データに対して、適切な部分構造を定義する
    - 木なら部分木、グラフならパス
  - カーネル関数の計算は、共通部分構造の個数を数える問題になる
    - 再帰構造を利用して、内積だけを効率的に計算



# 順序木に対するカーネル関数の設計 を行いました

- 順序木カーネルでは、部分構造を部分木によって定義する
- 難しいところ：部分木は指数個ありうる  
← 解決法：数え上げの計算を動的計画法によって再帰計算することで計算可能になる



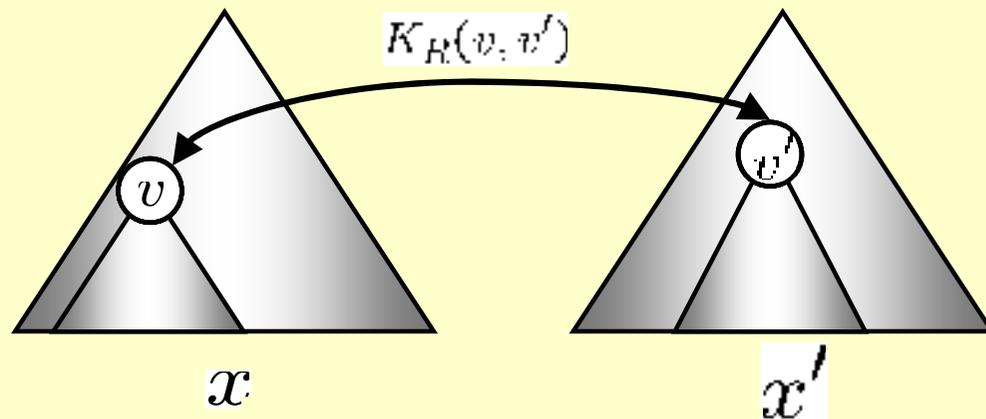
H. Kashima and T. Koyanagi: Kernels for Semi-Structured Data, In Proc. 19th International Conference on Machine Learning (ICML), 2002  
鹿島 久嗣, 坂本 比呂志, 小柳 光生: 木構造データに対するカーネル関数の設計と解析, 人工知能学会論文誌, Vol.21, No.1, 2006

# 順序木カーネルの計算は、再帰によって効率的に行えます

- 特徴ベクトルを、部分構造の集合として部分木をつかって構成
  - 特徴ベクトル  $\Phi(x) = (\begin{matrix} \text{C} \\ | \\ \text{D} \end{matrix}$  の出現回数,  $\begin{matrix} \text{A} \\ / \quad \backslash \\ \text{C} \quad \text{B} \quad \text{B} \end{matrix}$  の出現回数, ...)
- カーネル関数  $K$  は  $v$  と  $v'$  を根に持つ共通部分木の個数  $K_R$  の和としてかける

$$K(x, x') = \sum_{v \in V} \sum_{v' \in V'} K_R(v, v')$$

$$K_R(v, v') = \sum_{s \in S_v(x)} \sum_{s' \in S_{v'}(x')} \delta(s, s')$$



$v$  を根に持つ部分木の集合

$s$  と  $s'$  が同じなら 1

- $K_R$  は「 $(v, v')$  についてボトムアップ」 & 「 $(v, v')$  の子について左から右」の2重ループの動的計画法によって  $O(|V||V'|)$  で再帰計算

$(v, v')$  について

$$K^R(v, v') = \delta(\ell(v), \ell(v')) \cdot \bar{K}_{v, v'}^R(\#ch(v), \#ch(v'))$$

$(v, v')$  の子同士について

$$\begin{aligned} \bar{K}_{v, v'}^R(i, j) = & \bar{K}_{v, v'}^R(i-1, j) + \bar{K}_{v, v'}^R(i, j-1) - \bar{K}_{v, v'}^R(i-1, j-1) \\ & + \bar{K}_{v, v'}^R(i-1, j-1) \cdot K^R(ch(v, i), ch(v', j)) \end{aligned}$$

# 順序木カーネルの計算は、再帰によって効率的に行えます

- 特徴ベクトルを、部分構造の集合として部分木たちによって構成
  - 特徴ベクトル  $\Phi(x) = (\begin{matrix} C \\ D \end{matrix})$

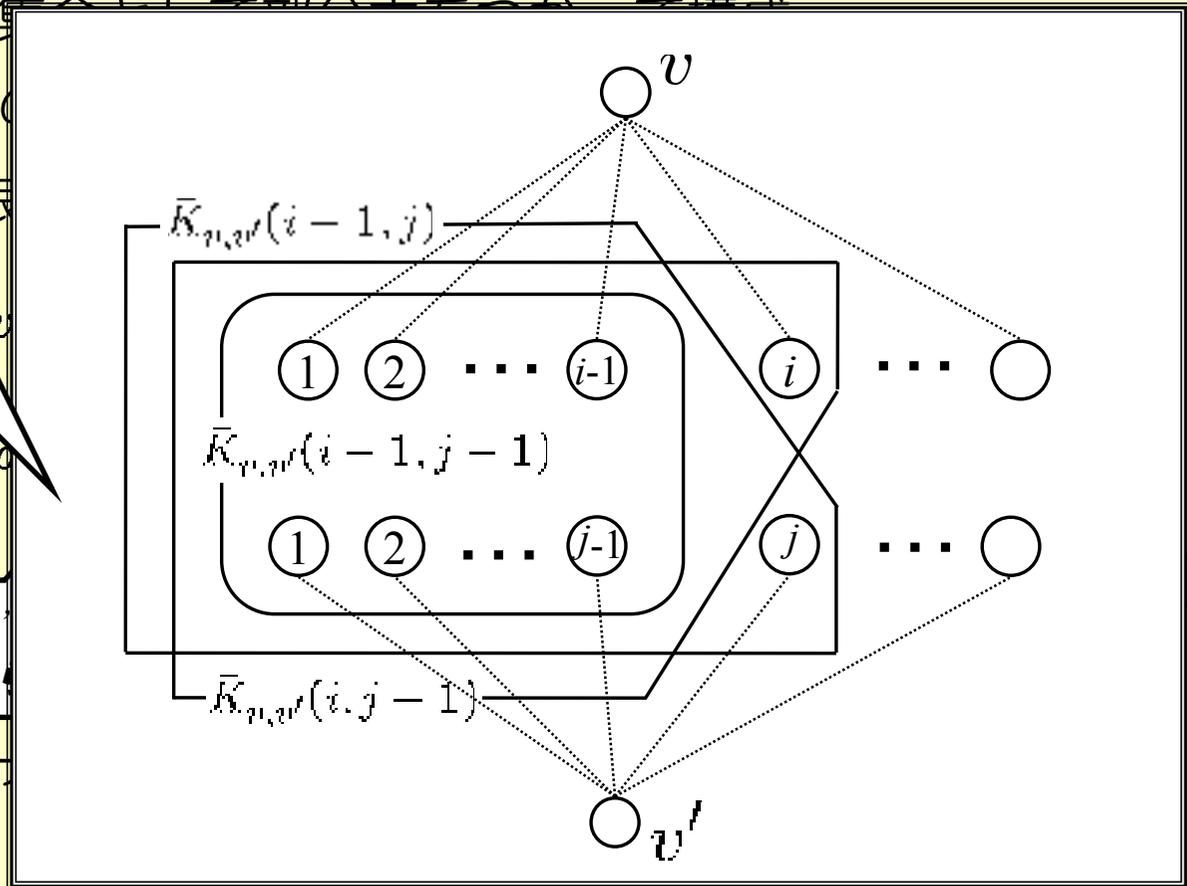
- カーネル

$$K(x, x')$$

$$K_R(v, v')$$

**(v, v') の子同士のマッチングを全て数え上げる**

$$s \in S_v(x) \quad s' \in S_{v'}(x')$$



**v を根に持つ部分木の集合**

**s と s' 同じな**

- $K_R$  は「(v, v') についてボトムアップループの動的計画法によって

**(v, v') について**

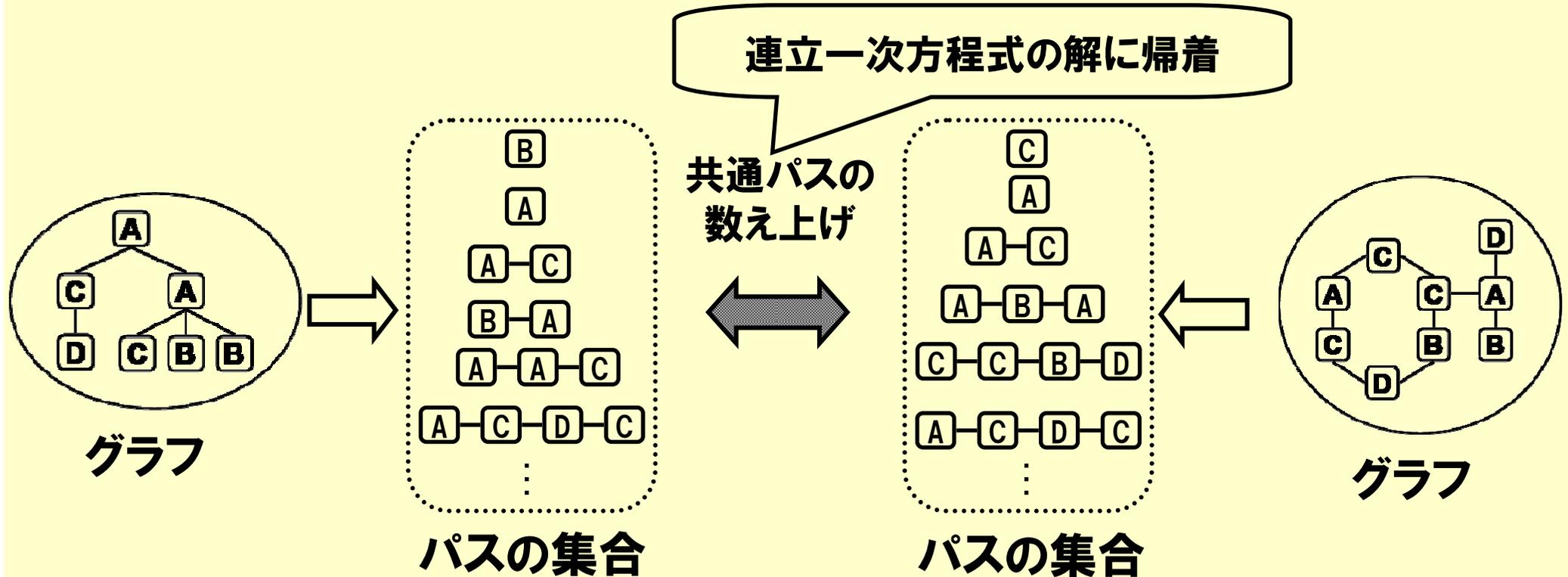
$$K^R(v, v') = \delta(\ell(v), \ell(v')) \cdot \bar{K}_{v, v'}^R(\#ch(v), \#ch(v'))$$

**(v, v') の子同士について**

$$\begin{aligned} \bar{K}_{v, v'}^R(i, j) = & \bar{K}_{v, v'}^R(i-1, j) + \bar{K}_{v, v'}^R(i, j-1) - \bar{K}_{v, v'}^R(i-1, j-1) \\ & + \bar{K}_{v, v'}^R(i-1, j-1) \cdot K^R(ch(v, i), ch(v', j)) \end{aligned}$$

# グラフに対するカーネル関数の設計を行いました

- グラフカーネルでは、部分構造を、グラフ上のランダムウォークによって取り出されるパスとして定義する
- 難しいところ：パスは無限個ありうる  
← 解決法：数え上げの計算を連立一次方程式に帰着することで計算可能になる



H. Kashima, K. Tsuda and A. Inokuchi: Marginalized Kernels Between Labeled Graphs, In Proc. 20th International Conference on Machine Learning (ICML), 2003  
Hisashi Kashima, Koji Tsuda and Akihiro Inokuchi: Kernels for Graphs, In Kernel Methods in Computational Biology, MIT Press, 2004

# グラフ・カーネルの計算は、連立方程式に帰着され、効率的に行えます

- ランダム・ウォークによる無限個のパスによって特徴ベクトルを構成
  - 特徴ベクトル  $\Phi(x) = ( \boxed{A-C}$  の出現回数  $\times \lambda^2$ ,  $\boxed{A-A-C}$  の出現回数  $\times \lambda^3$ , ...)
  - 発散しないように、パスの長さによって指数的に減衰される
- 再帰的表現に持ち込むことで効率的に計算可能になる

-  $S_v(x)$ : ノード  $v$  で終わるパスの集合

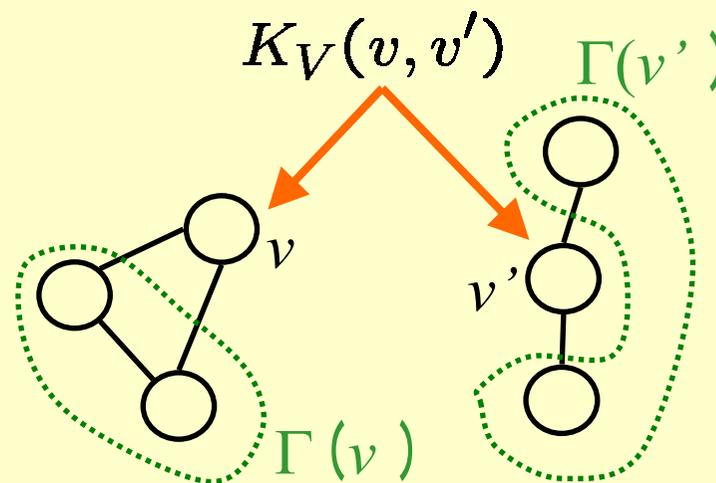
- ノード対  $(v, v')$  で終わる同じラベル列をもつパスの個数  $K_V$  の和に分解できる

$$K(x, x') = \sum_{v \in V} \sum_{v' \in V'} K_V(v, v')$$

$$K_V(v, v') = \sum_{s \in S_v(x)} \sum_{s' \in S_{v'}(x')} \lambda^{|s|} \lambda^{|s'|} \delta(s, s')$$

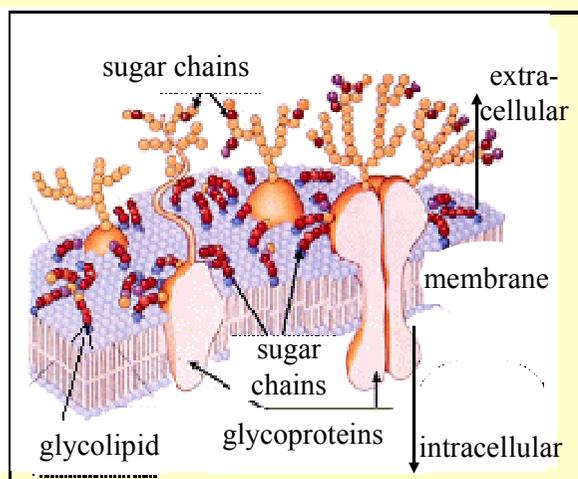
-  $K_V(v, v')$  は近隣ノード同士の  $K_V$  の和によって再帰的に書けるので、連立方程式を解けばカーネルが計算できる

$$K_V(v, v') = \lambda^2 \delta(\ell(v), \ell(v')) \left( 1 + \sum_{\tilde{v} \in \Gamma(v)} \sum_{\tilde{v}' \in \Gamma(v')} \lambda^2 K_V(\tilde{v}, \tilde{v}') \right)$$

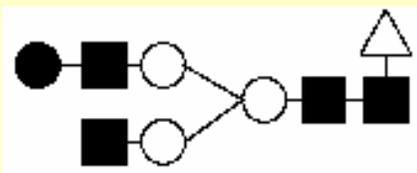


## 応用：木カーネルによる糖鎖の構造分類

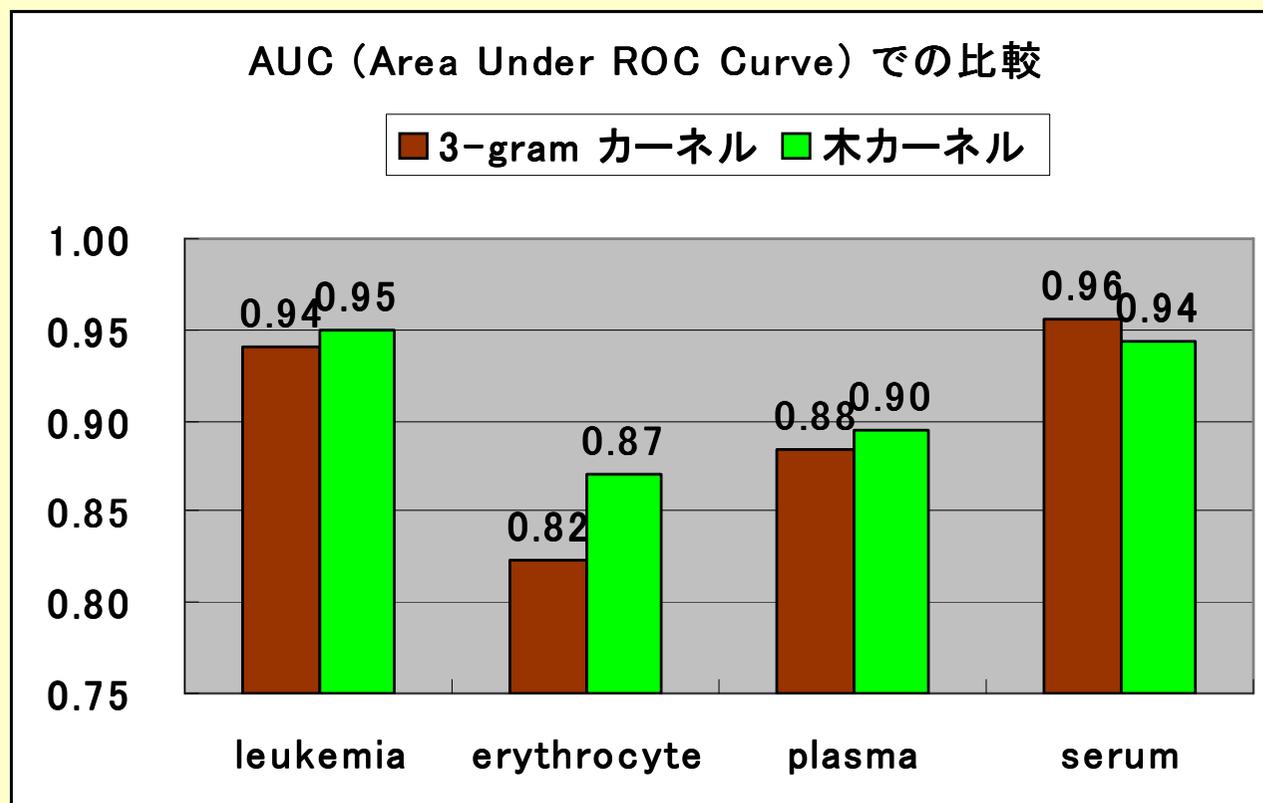
- 糖鎖の構造から組織を予測する問題に適用し、単純な方法（3-gramに基づく方法）を若干上回る性能が得られた



### 糖鎖の木構造表現

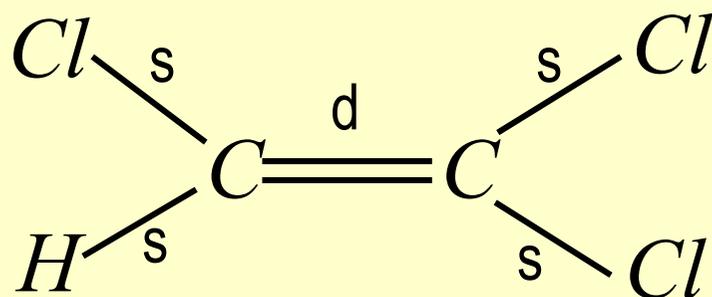


■	N-acetylglucosamine (GlcNAc)	△	Fucose (Fuc)
○	Mannose (Man)	▲	Glucose (Glu)
●	Galactose (Gal)		



## 応用： グラフカーネルによる化合物の毒性予測

- 指数時間の他手法（パターンマイニング）に匹敵する性能が得られた



化合物のグラフ表現

### パターンマイニング

MinSup	MM	FM	MR	FR
0.5%	60.1%	57.6%	61.3%	<b>66.7%</b>
1 %	<b>61.0%</b>	<b>61.0%</b>	<b>62.8%</b>	63.2%
3 %	58.3%	55.9%	60.2%	63.2%
5 %	60.7%	55.6%	57.3%	63.0%
10 %	58.9%	58.7%	57.8%	60.1%
20%	<b>61.0%</b>	55.3%	56.1%	61.3%

μ <sub>1</sub> (G)	MM	FM	MR	FR
0.1	62.8%	61.6%	<b>58.4%</b>	<b>66.1%</b>
0.2	63.4%	<b>63.4%</b>	54.9%	64.1%
0.3	63.1%	62.5%	54.1%	63.2%
0.4	62.8%	61.9%	54.4%	65.8%
0.5	64.0%	61.3%	56.1%	64.4%
0.6	<b>64.3%</b>	61.9%	56.1%	63.0%
0.7	64.0%	61.3%	56.7%	62.1%
0.8	62.2%	61.0%	57.0%	62.4%
0.9	62.2%	59.3%	57.0%	62.1%

### グラフカーネル

## これらの研究はその後、さまざまな発展を遂げています

---

- 順序木カーネル
  - 曖昧マッチングのとりこみ [久保山ら, 2006]
  - 部分構造をパスに限定した高速化 [Kuboyama et al., 2006, 2007]
  - 糖鎖構造分類への応用 [Kuboyama et al., 2006]
- グラフカーネル
  - ランダムウォークの設計による高精度化 [Mahe et al., 2004]
  - 行列計算を利用した各種高速化 [Vishwanathan et al., 2006]
  - Cyclic Pattern [Horvath et al., 2004]、Shortest Path [Borgwardt & Kriegel., 2005]などによる近似による高速化
  - タンパク質立体構造分類 [Borgwardt et al., 2005]、画像分類への応用
- より複雑な問題への適用
  - 構造データのラベル付け問題への適用 [Kashima et al., 2004]

## 前半のまとめ： 内部構造（木、グラフ）を扱うための、 カーネル法について紹介しました

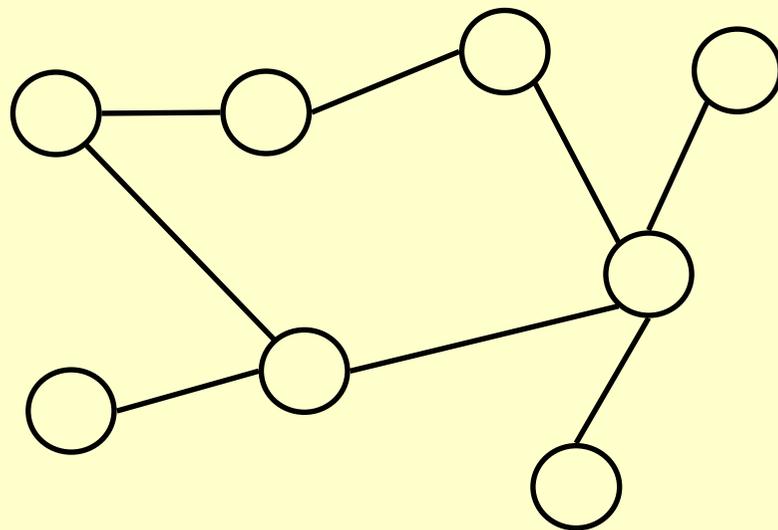
- カーネル法は、学習とカーネル関数を分離して考えることのできる便利な枠組みです
- 構造データのカーネル関数を、表現力と計算の速さのバランスをとりながら、うまく設計しました
- 部分構造の再帰性を利用した、効率のよい計算方法を与えました
  - [研究成果 1] ラベル付順序木カーネル
    - 順序木に対するカーネルを、部分木を用いて定義しました
    - 動的計画法を用いた効率的なアルゴリズムを設計しました
  - [研究成果 2] グラフ・カーネル
    - グラフに対するカーネルを、ランダムウォークによって生成されるパスを用いて定義しました
    - 連立一次方程式に帰着することで効率的なアルゴリズムを設計しました

---

## 外部構造の扱いについて

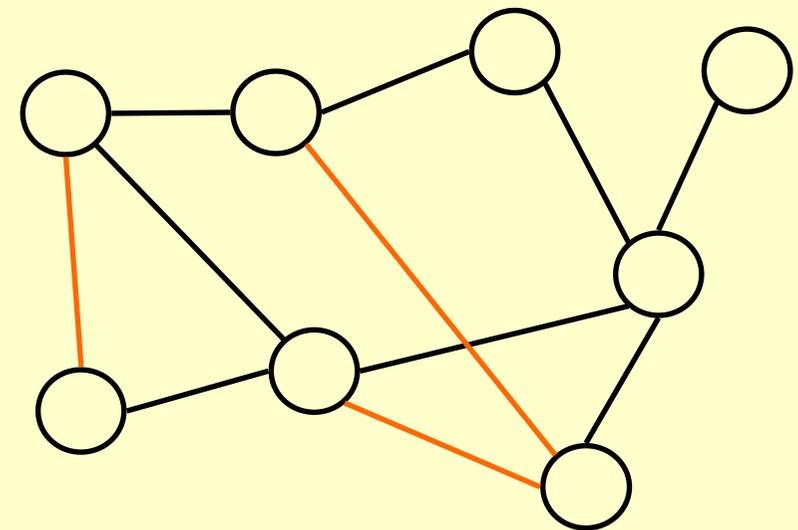
# 外部構造の学習問題として、 リンク予測の問題についてお話します

- リンク予測問題：部分的に観測されているネットワーク構造から、残りの構造を推定する問題
  - 「リンクマイニング」の主要タスクのひとつ
  - 例：
    - 生体ネットワークの構造予測
    - SNSにおける友人の推薦
    - 協調フィルタリング



部分的に観測される構造

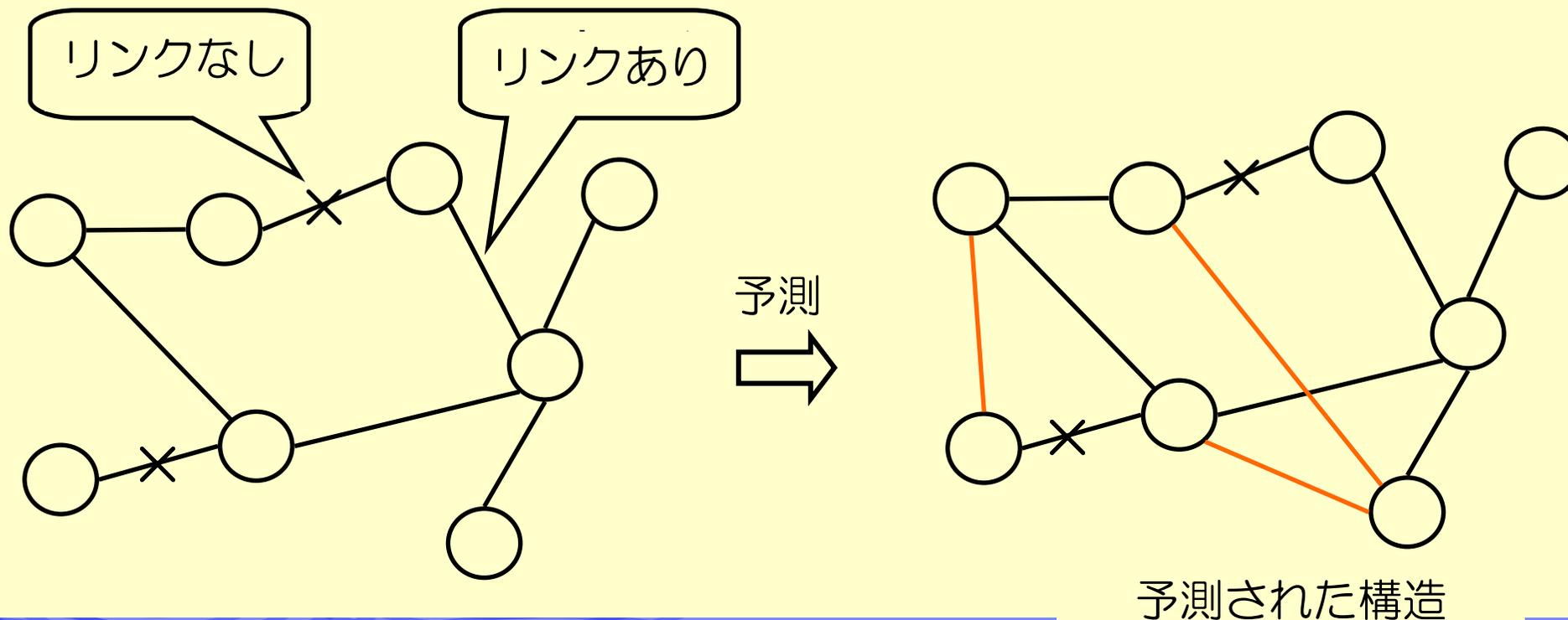
補完  
⇒



予測される構造

# 一般に、（教師つき）リンク予測問題は、 グラフ構造の補完問題として定義できます

- 入力：一部が欠けたネットワーク構造
  - リンクありのノードペア
  - リンクのないノードペア
- 出力：リンクの有無が未知のノードペアについてのリンク予測



# リンク予測の応用

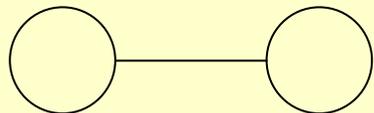
- 応用：
  - 生体ネットワークの構造予測
  - ソーシャルネットワークでの推薦や関係の予測
- ネットワーク構造をもったデータの例：

ネットワーク	ノード	リンク
WWW	Webページ	ハイパーリンク
社会ネットワーク	人 コミュニティ	友人関係 所属
生体ネットワーク	遺伝子 タンパク質	制御 相互作用

# リンク予測のひとつの捉え方は 「2つのノードのペアの教師つき分類問題」です

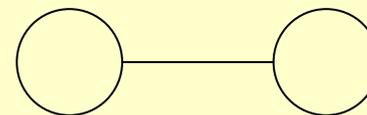
- 任意の2つのノードの間のリンクの強さを、予測する
  - 協調フィルタリング：ユーザーと、商品との購買関係を予測する
    - 買う=リンクあり、買わない=リンクなし
  - 生体ネットワーク予測：2つのたんぱく質の相互作用の有無を予測する
    - 相互作用あり=リンクあり、相互作用なし=リンクなし
- 正解（リンクの有無）は、いくつかのペアについては与えられる
  - これをもとに正解未知のペアについて、リンクの強さを予測する
  - これも教師つき学習の問題のひとつ

ユーザー A    商品 B



協調フィルタリング

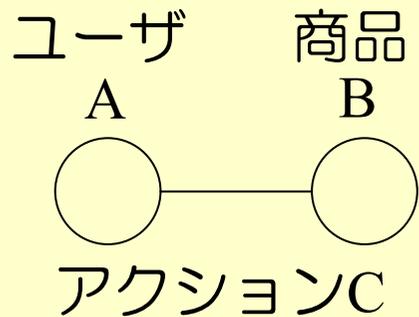
たんぱく質 A    たんぱく質 B



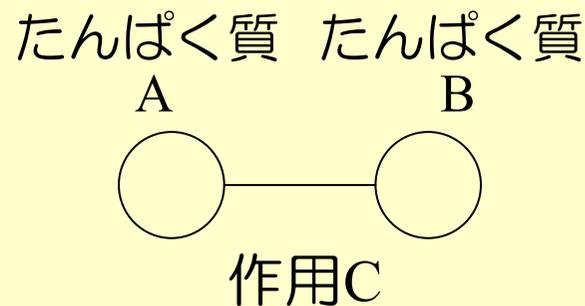
生体ネットワーク予測

# さらに、リンクの種類を考えた「複数タイプリンク予測」は、より詳細なモデル化を可能にします

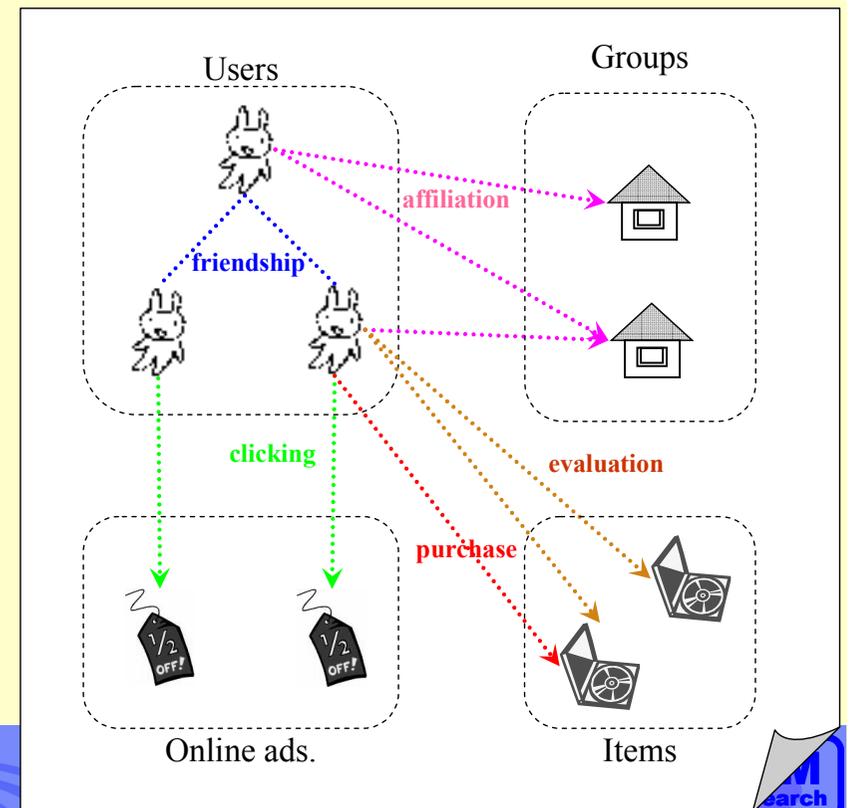
- 複数タイプリンク予測は、「リンクの種類」を考える
  - 協調フィルタリング：購買、評価、商品情報の閲覧、など
  - 生体ネットワーク予測：作用の種類や、作用が起こる環境、など
- 異なるタイプのリンク間の相関を利用することで、予測精度があがる可能性がある



協調フィルタリング



生体ネットワーク予測



## このあとの話の流れ

---

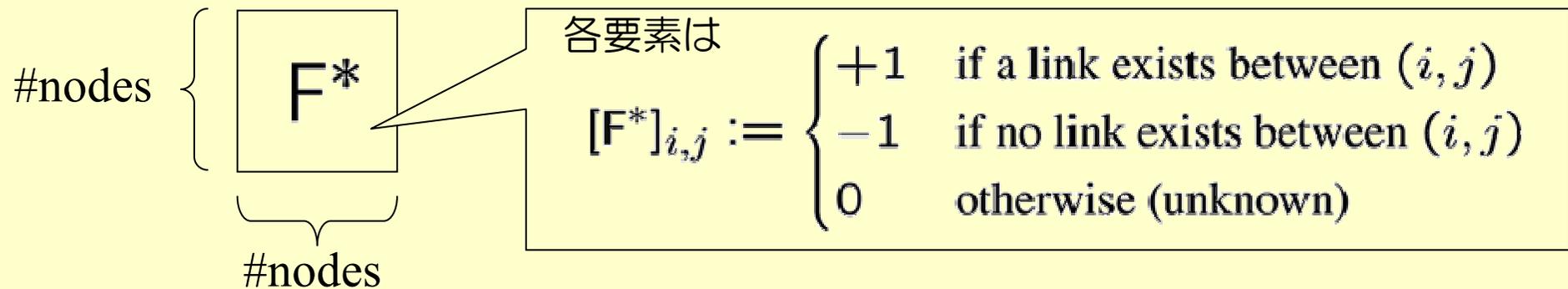
- 「複数タイプリンク予測」の定式化
  - 外部情報（類似度情報）を用いたテンソル補完の問題として定式化する
  
- 我々のアプローチ：半教師付学習「ラベル伝播」の適用
  - 扱いやすい類似度の定義
  - 共役勾配法に基づく効率的な解法

Hisashi Kashima, Tsuyoshi Kato, Yoshihiro Yamanishi, Masashi Sugiyama and Koji Tsuda:

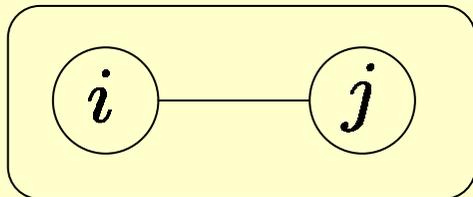
Link Propagation: A Supervised Learning Algorithm for Link Prediction, In Proc. 2009 SIAM Conference on Data Mining (SDM), 2009.

# まず、単一タイプのリンク予測問題が、 行列補完の問題として捉えられることを見ます

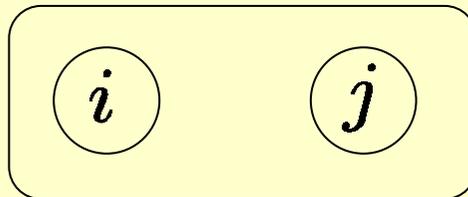
- ネットワーク構造の既知部分が、隣接行列  $F^*$  で与えられている
  - 既知部分は +1 か -1 で、未知部分は 0
- 目標：未知部分 (0) を [-1, +1] の値で（リンクの確信度に応じて）埋める



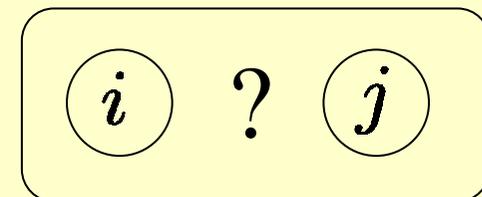
$$[F^*]_{i,j} = +1$$



$$[F^*]_{i,j} = -1$$

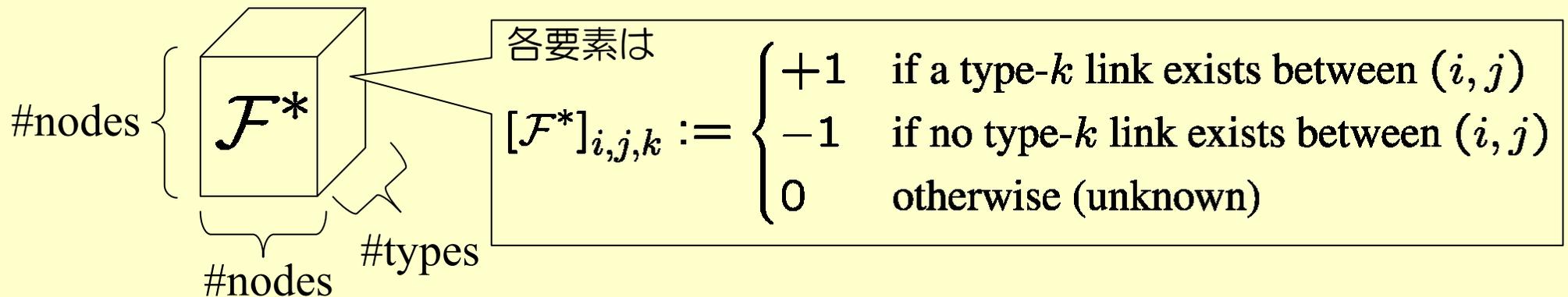


$$[F^*]_{i,j} = 0$$

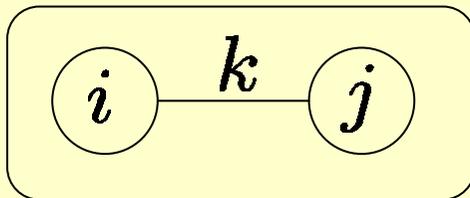


# 複数タイプリンク予測問題は、 3階のテンソルの補完問題として捉えられます

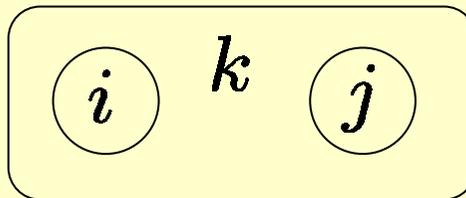
- ネットワーク構造の既知部分が、3階のテンソル  $\mathcal{F}^*$  で与えられる
  - 既知部分は +1 か -1 で、未知部分は 0
- 目標：未知部分 (0) を [-1, +1] の値で（リンクの確信度に応じて）埋める



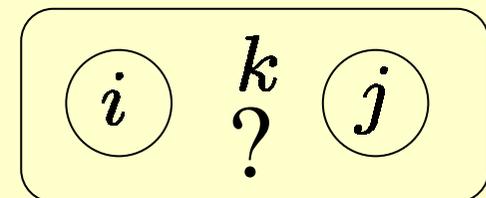
$$[\mathcal{F}^*]_{i,j,k} = +1$$



$$[\mathcal{F}^*]_{i,j,k} = -1$$



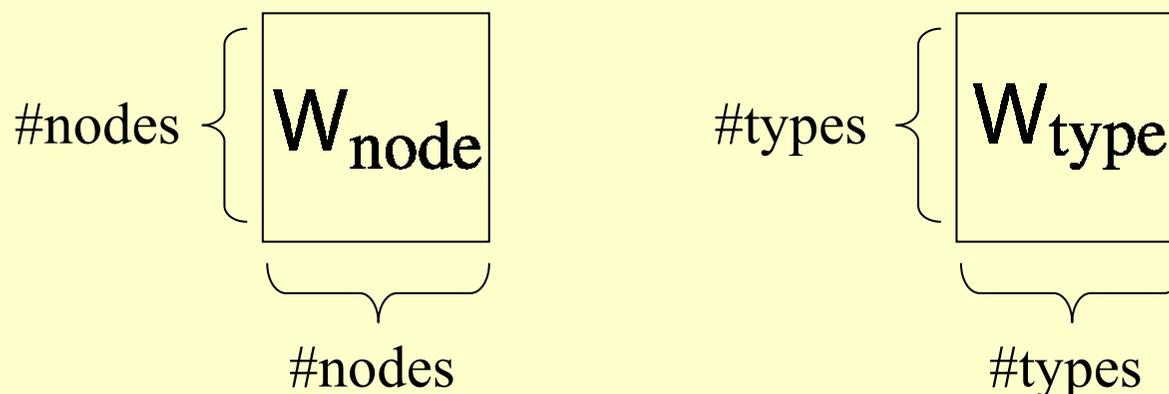
$$[\mathcal{F}^*]_{i,j,k} = 0$$



仮定：今回は、補助情報として、ノード間の類似度や、リンクタイプ間の類似度が与えられているとします

- リンクの有無の情報のほかに、事前知識としての類似度が与えられている場合がしばしばある
- 今回は、ノード間の類似度行列 と リンクタイプ間の類似度行列が与えられているとする

– カーネル関数に相当

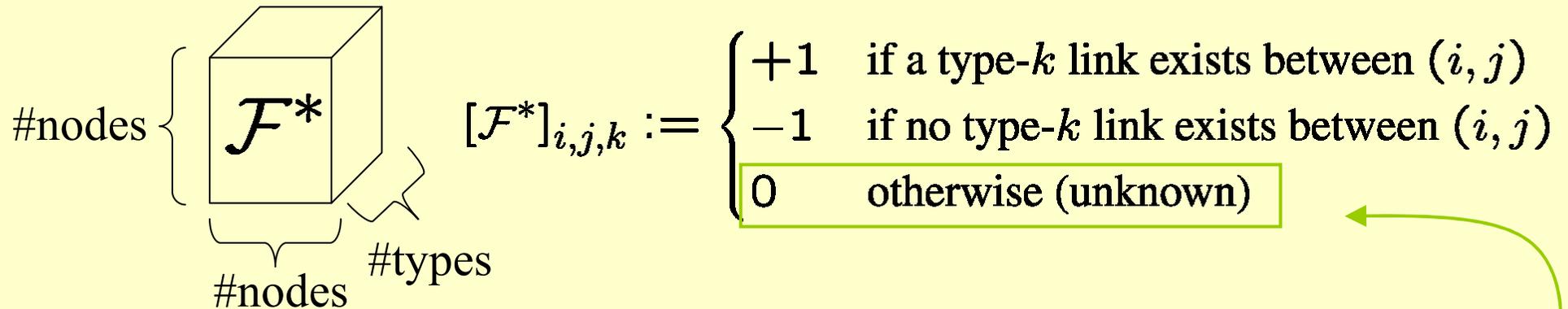


- たとえば、バイオインフォマティクスでは、
  - $W_{\text{node}}$  は、タンパク質の配列、遺伝子発現、系統発生、局在部位、などの類似度を
  - $W_{\text{type}}$  は、2つのリンクタイプの共起度合い

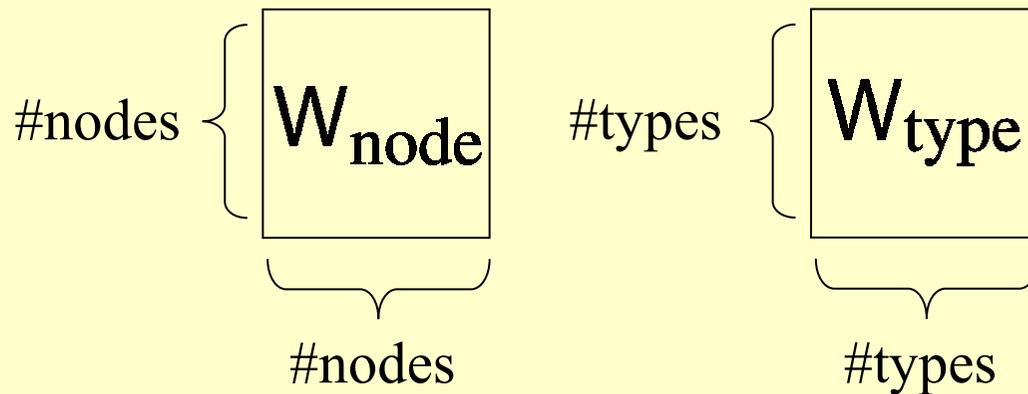
# 問題のまとめ：複数タイプリンク予測問題

- 入力:

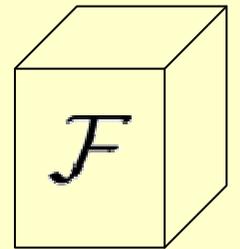
- ネットワークの既知部分を表した3階テンソル



- ノード間、タイプ間の類似度行列

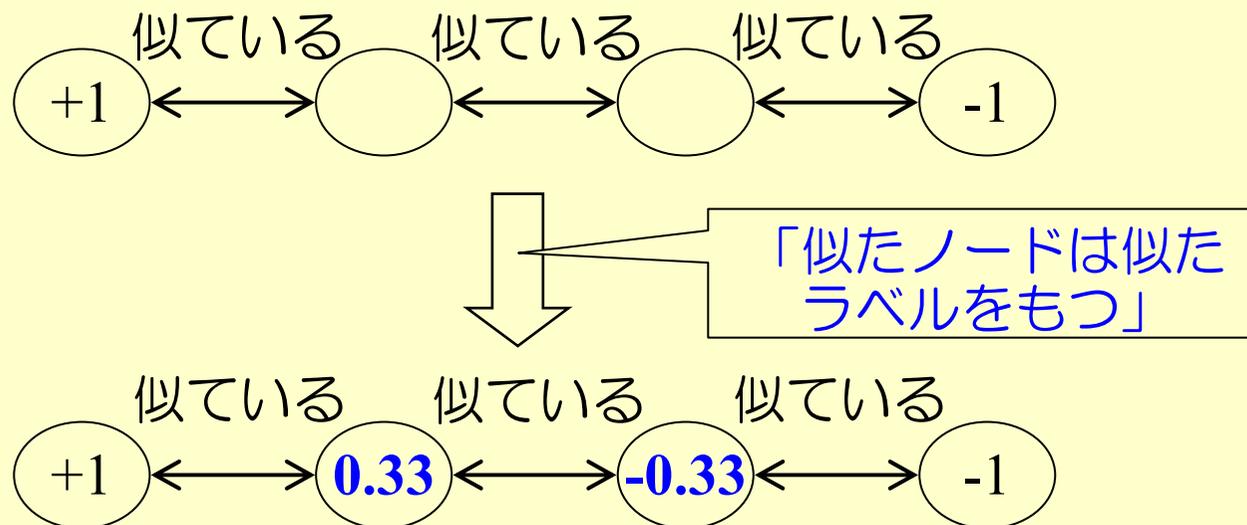


- 出力: 構造が未知の部分のリンク強度を表した3階テンソル



# この問題は、半教師付きの予測問題なので、半教師付きの代表的手法「ラベル伝播」を用いることにします

- この問題は、リンク未知のノードペアが予めわかっているので、「半教師付き」の問題として捉えられる
  - テストデータの情報を利用できる
- 半教師付き学習の代表的手法「ラベル伝播」を用いる
  - 元々は、ノードの分類（ノードペアではなく）のための手法
  - 「似たノードは、似たラベルをもつ」の法則を用いる



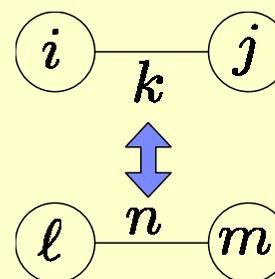
## 目的関数の定義: ラベル伝播を、ノード「ペア」に適用します

- ラベル伝播を、3つ組み  $(i, j, k)$  に対して適用 (ノードペアとリンクタイプ)
- 目的関数  $J(\mathcal{F})$  を最小化する、3階テンソル  $\mathcal{F}$  を求める
  - 1項目: 「似ている3つ組みは、近いリンク強度をもつ」 (リンク伝播)
    - $\tilde{w}_{ijk,lmn} > 0$  : 2つの3つ組の間の類似度
  - 2項目: 「リンク強度の予測値は、既知の構造に近づける」

$J(\mathcal{F}) :=$

$$\frac{\sigma}{2} \sum_{i,j,k,l,m,n} \tilde{w}_{ijk,lmn} ([\mathcal{F}]_{i,j,k} - [\mathcal{F}]_{l,m,n})^2 + \frac{1}{2} \sum_{(i,j,k)} ([\mathcal{F}]_{i,j,k} - [\mathcal{F}^*]_{i,j,k})^2$$

2つの3つ組み  $(i,j,k)$  と  $(l,m,n)$   
の間の類似度  
(後で定義する)



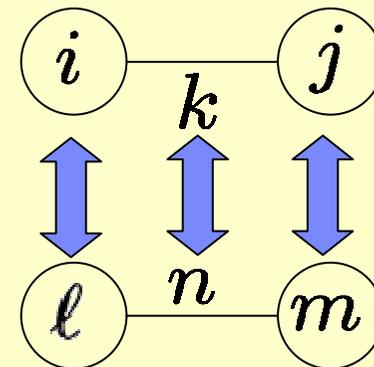
# 3つ組同士の類似度を、与えられた類似度行列の「クロネッカー積」か「クロネッカー和」によって定義することにします

- 3つ組み同士の類似度を、もともとの類似度行列から定義

## 1. クロネッカー積類似度 $\tilde{W} := W_{\text{type}} \otimes W_{\text{node}} \otimes W_{\text{node}}$

- 「対応するノード同士が全て似ているなら、3つ組み同士も似ている」

$$\left[ \tilde{w}_{ijk,lmn} := [W_{\text{node}}]_{i,\ell} \cdot [W_{\text{node}}]_{j,m} \cdot [W_{\text{type}}]_{k,n} \right]$$

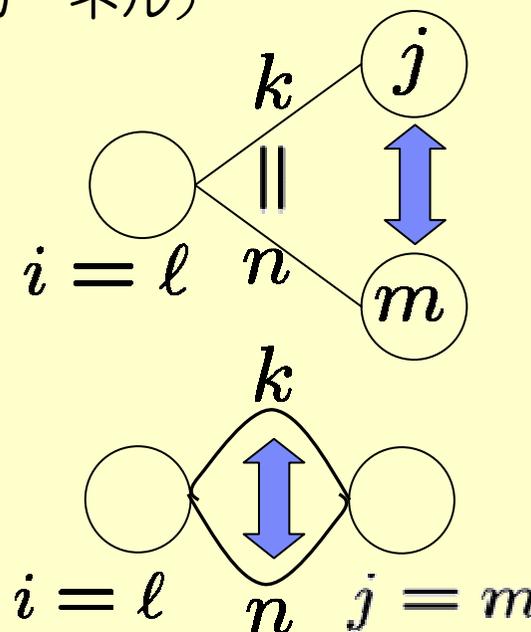


- 特徴空間のクロネッカー積における内積（ペアワイズカーネル）

## 2. クロネッカー和類似度 $\tilde{W} := W_{\text{type}} \oplus W_{\text{node}} \oplus W_{\text{node}}$

- 「2つが共通で、もう1つが似ているなら、3つ組み同士も似ている」

$$\left[ \tilde{w}_{ijk,lmn} := \begin{aligned} & [W_{\text{node}}]_{i,\ell} \cdot \delta(j = m) \cdot \delta(k = n) \\ & + \delta(i = \ell) \cdot [W_{\text{node}}]_{j,m} \cdot \delta(k = n) \\ & + \delta(i = \ell) \cdot \delta(j = m) \cdot [W_{\text{type}}]_{k,n} \end{aligned} \right]$$



# 単純にラベル伝播を適用するだけでは計算量的な問題があります

- 目的関数を行列を使って書き直すと

$$J(\mathcal{F}) = \frac{\sigma}{2} \text{vec}(\mathcal{F})^\top \tilde{\mathbf{L}} \text{vec}(\mathcal{F}) + \frac{1}{2} \|\text{vec}(\mathcal{F}) - \text{vec}(\mathcal{F}^*)\|_2^2$$

–  $\tilde{\mathbf{L}}$  はラプラシアン行列  $\tilde{\mathbf{L}} := \bar{\mathbf{D}} - \bar{\mathbf{W}}$   $\tilde{w}_{ijk,lmn}$  の行列表現

- 結局、以下の連立方程式を解くことによって解が求まる

$$(\sigma \tilde{\mathbf{L}} + \mathbf{I}) \text{vec}(\mathcal{F}) = \text{vec}(\mathcal{F}^*)$$

$(\#\text{nodes}^2 \cdot \#\text{types}) \times (\#\text{nodes}^2 \cdot \#\text{types})$  の  
巨大な行列

$$\text{vec} \left( \begin{array}{|c|} \hline \text{red} \\ \text{green} \\ \text{blue} \\ \text{grey} \\ \hline \end{array} \right) = \left( \begin{array}{c} \text{red} \\ \text{blue} \\ \text{green} \\ \text{grey} \end{array} \right)$$

- しかし、連立方程式が大きすぎる...

# ひとまず、共役勾配法を適用してみるものの、 計算のボトルネックがあります

- テンソル版の共役勾配法

- 置き換え :  $A = (\sigma\tilde{L} + I)$ ,  $\mathbf{f} = \text{vec}(\mathcal{F})$  and  $\mathbf{f}^* = \text{vec}(\mathcal{F}^*)$

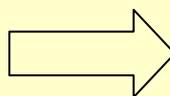
- $(\sigma\tilde{L} + I)$  が大きいいため、**vec(Q(t))** の計算部分がボトルネック

Conjugate gradient for  $A\mathbf{f} = \mathbf{f}^*$   
(standard)

```
1:  $\mathbf{f}(0) := \mathbf{f}^*$ 
2:  $\mathbf{q}(0) := A\mathbf{f}(0)$ 
3:  $\mathbf{r}(0) := \mathbf{f}^* - \mathbf{q}(0)$ , and  $\mathbf{p}(0) := \mathbf{r}(0)$ 
4: for  $t = 0, 1, 2, \dots$  do
5:    $\mathbf{q}(t) := A\mathbf{p}(t)$ 
6:    $\alpha(t) := \frac{\langle \mathbf{r}(t), \mathbf{p}(t) \rangle}{\langle \mathbf{p}(t), \mathbf{q}(t) \rangle}$ 
7:    $\mathbf{f}(t+1) := \mathbf{f}(t) + \alpha(t)\mathbf{p}(t)$ 
8:    $\mathbf{r}(t+1) := \mathbf{r}(t) - \alpha(t)\mathbf{q}(t)$ 
9:    $\beta(t) := \frac{\|\mathbf{r}(t+1)\|_2^2}{\|\mathbf{r}(t)\|_2^2}$ 
10:  if  $\frac{\|\mathbf{r}(t+1)\|_2}{\|\mathbf{r}(0)\|_2} < \epsilon$ , return  $\mathbf{f}(t+1)$ 
11:   $\mathbf{p}(t+1) := \mathbf{r}(t+1) + \beta(t)\mathbf{p}(t)$ 
12: end for
```

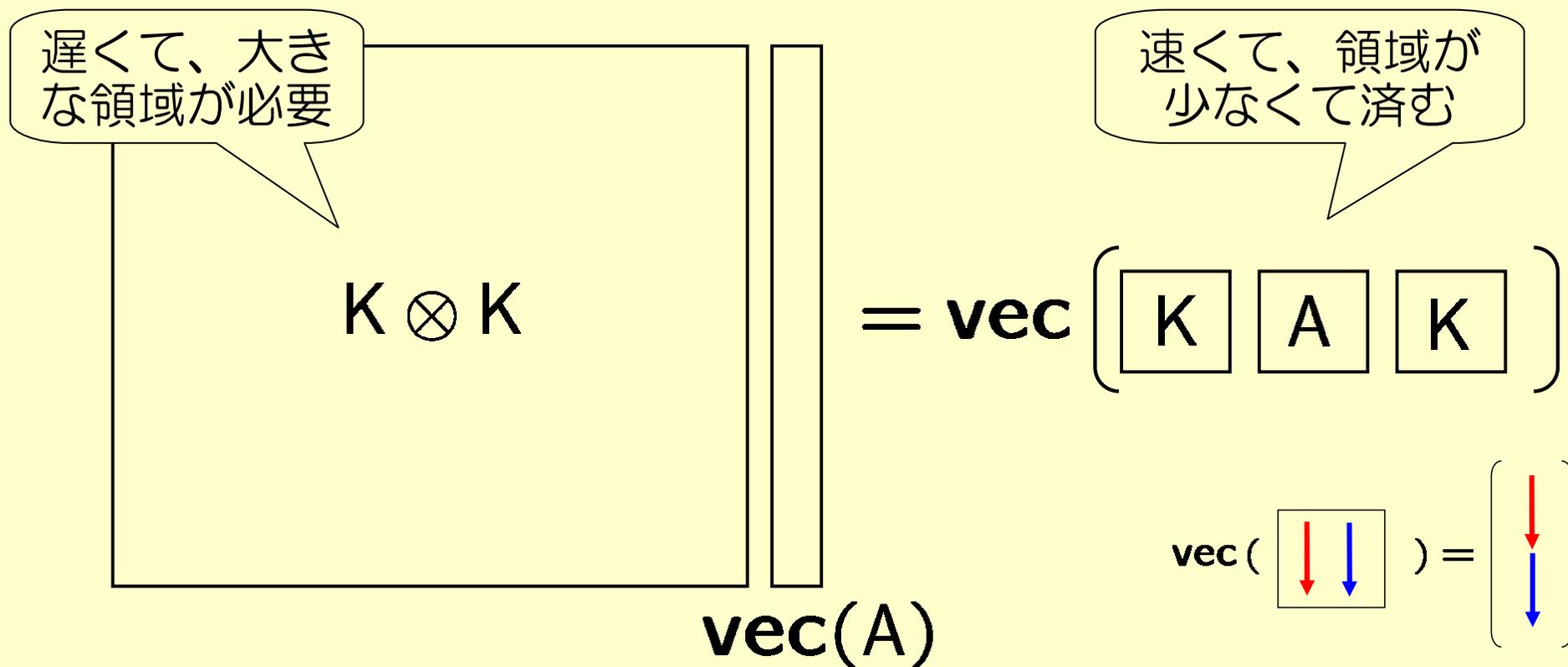
Conjugate gradient for  $(\sigma\tilde{L} + I)\text{vec}(\mathcal{F}) = \text{vec}(\mathcal{F}^*)$   
(tensorized)

```
1:  $\mathcal{F}(0) := \mathcal{F}^*$ 
2:  $\text{vec}(\mathcal{Q}(0)) := (\sigma\tilde{L} + I)\text{vec}(\mathcal{F}(0))$ 
3:  $\mathcal{R}(0) := \mathcal{F}^* - \mathcal{Q}(0)$ , and  $\mathcal{P}(0) := \mathcal{R}(0)$ 
4: for  $t = 0, 1, 2, \dots$  do
5:   $\text{vec}(\mathcal{Q}(t)) := (\sigma\tilde{L} + I)\text{vec}(\mathcal{P}(t))$ 
6:   $\alpha(k) := \frac{\langle \mathcal{R}(k), \mathcal{P}(k) \rangle}{\langle \mathcal{P}(k), \mathcal{Q}(t) \rangle}$ 
7:   $\mathcal{F}(k+1) := \mathcal{F}(k) + \alpha(k)\mathcal{P}(k)$ 
8:   $\mathcal{R}(k+1) := \mathcal{R}(k) - \alpha(k)\mathcal{Q}(k)$ 
9:  if  $\frac{\|\mathcal{R}(k+1)\|_2}{\|\mathcal{R}(0)\|_2} < \epsilon$ , return  $\mathcal{F}(k+1)$ 
10:  if  $\frac{\|\mathbf{r}(t+1)\|_2}{\|\mathbf{r}(0)\|_2} < \epsilon$ , return  $\mathbf{f}(t+1)$ 
11:   $\mathcal{P}(k+1) := \mathcal{R}(k+1) + \beta(k)\mathcal{P}(k)$ 
12: end for
```



## ここで、線形代数の標準的な公式が「非常に」役に立ちます

- 「行列のクロネッカー積」と「ベクトル化された行列」の積についての公式
- 左辺より、右辺がずっと効率的に計算できる



## 共役勾配法を高速化します

- 計算したいのは、 $\text{vec}(Q(t)) := (\sigma \tilde{L} + I)\text{vec}(P(t))$ 、ただし

- クロネッカー積類似度を使う場合

$$\tilde{L} = D_{\text{type}} \otimes D_{\text{node}} \otimes D_{\text{node}} - W_{\text{type}} \otimes W_{\text{node}} \otimes W_{\text{node}}$$

- クロネッカー和類似度を使う場合

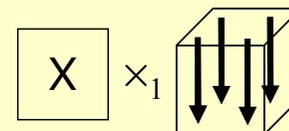
$$\begin{aligned} \tilde{L} &= D_{\text{type}} \oplus D_{\text{node}} \oplus D_{\text{node}} - W_{\text{type}} \oplus W_{\text{node}} \oplus W_{\text{node}} \\ &= L_{\text{type}} \oplus L_{\text{node}} \oplus L_{\text{node}} \end{aligned}$$

Tensor multiplication  $\times_k X$  multiplies the  $k$ -th fiber with a matrix  $X$

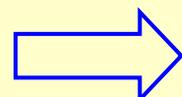
- 前述の公式（のテンソル版）

$$(X \otimes Y \otimes Z)\text{vec}(B) = \text{vec}(B \times_1 Z \times_2 Y \times_3 X)$$

$$(X \oplus Y \oplus Z)\text{vec}(B) = \text{vec}(B \times_1 Z + B \times_2 Y + B \times_3 X)$$



$(\#\text{nodes}^2 \cdot \#\text{types}) \times (\#\text{nodes}^2 \cdot \#\text{types})$   
の巨大な行列

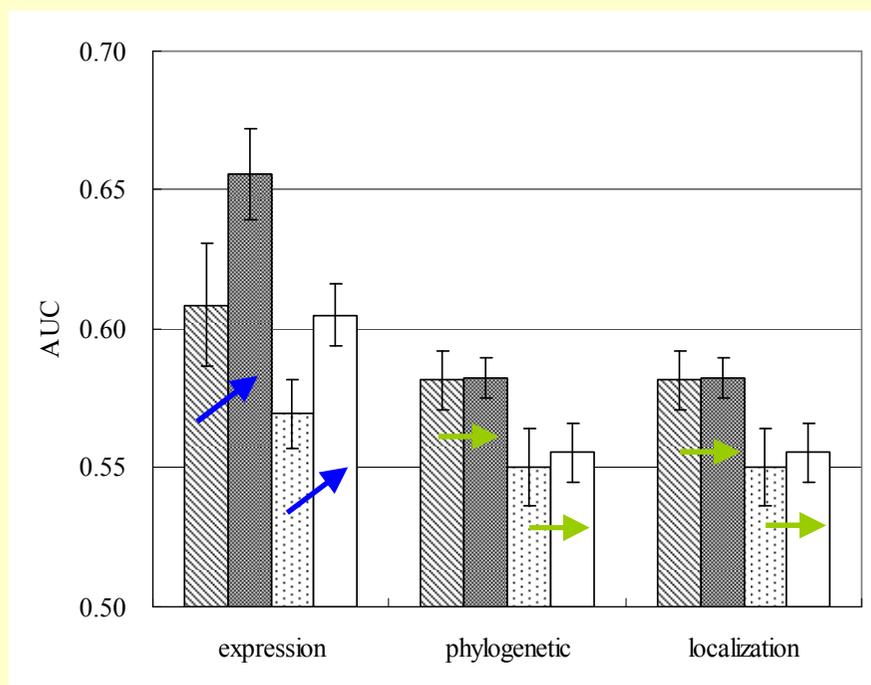


$\#\text{nodes} \times \#\text{nodes} \times \#\text{types}$  テンソル

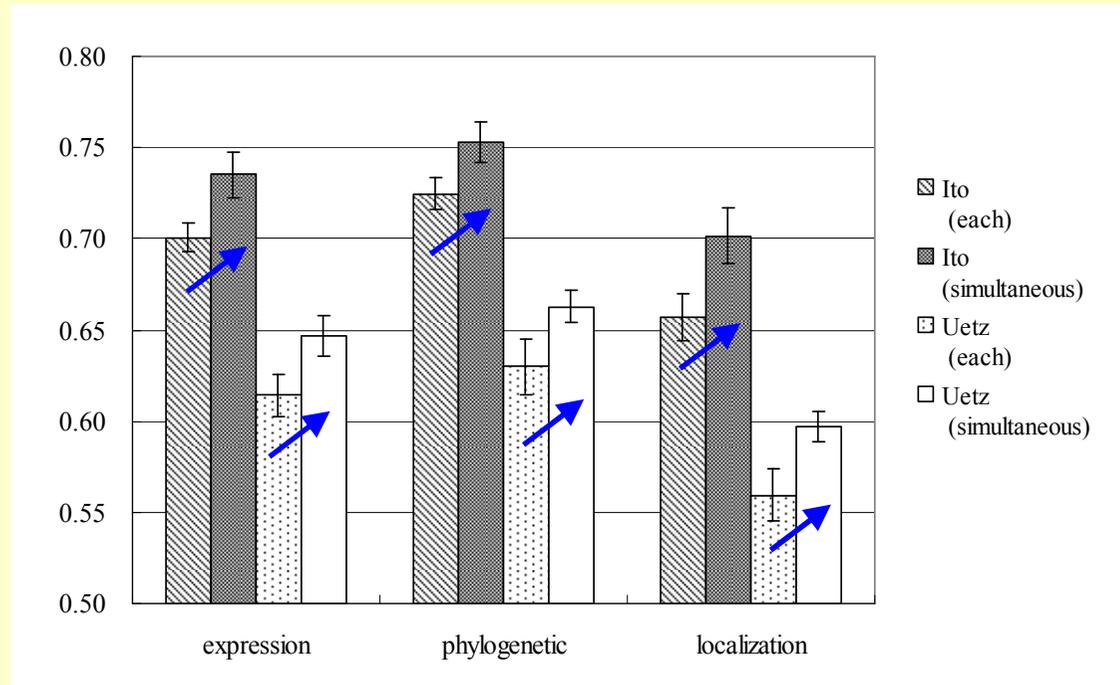
テンソルの掛け算で置き換えることで、  
時間もメモリも大幅に節約できる

## 2つの生体ネットワークの同時予測では、 同時予測によって予測精度が向上します

- 2つの研究室のタンパク質ネットワークを同時予測
  - ~1,500 ノード、700~900リンク、~150リンクを共有
- 同時予測によって性能が向上
- クロネッカー和類似度のほうが全体的に性能がよい



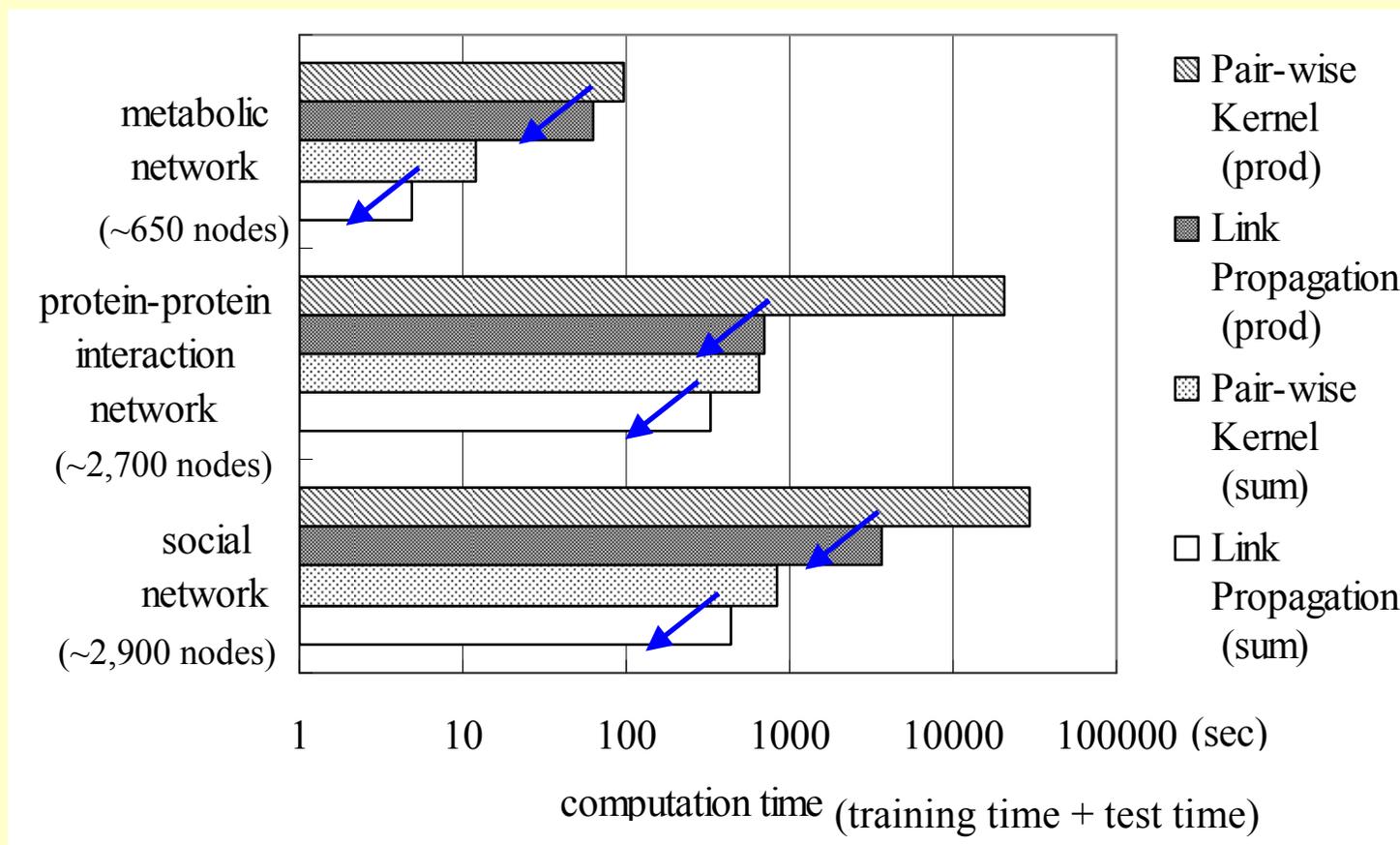
クロネッカー積類似度



クロネッカー和類似度

## 提案手法は、単一タイプのリンク予測において、既存手法（SVM）よりも遥かに高速です

- 提案手法は、同じ類似度をカーネル関数として用いたオンラインSVMより遥かに高速
- クロネッカー和のほうがクロネッカー積よりも高速

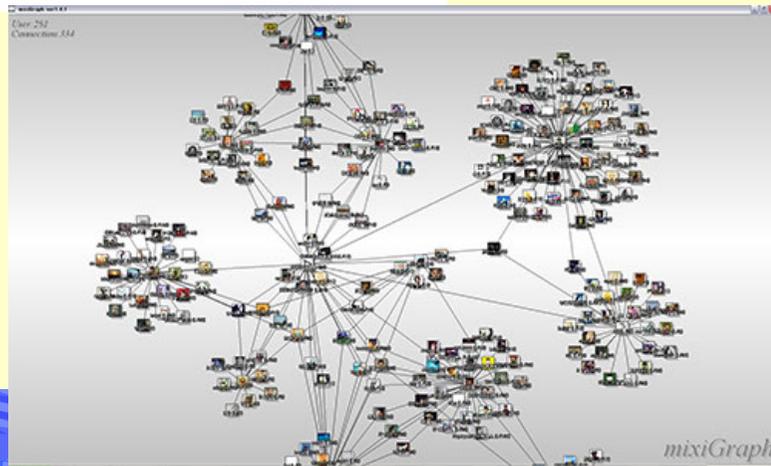
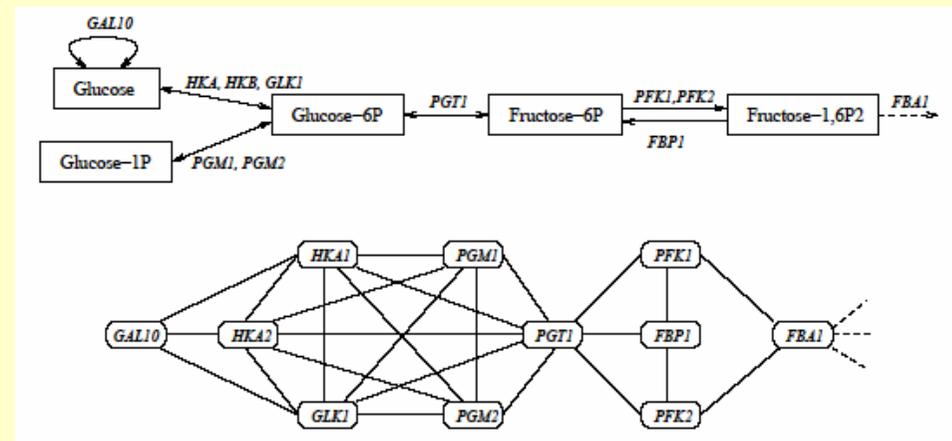
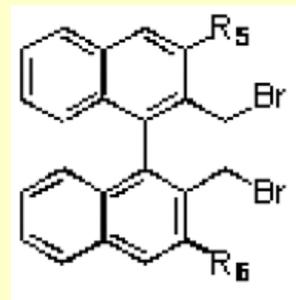
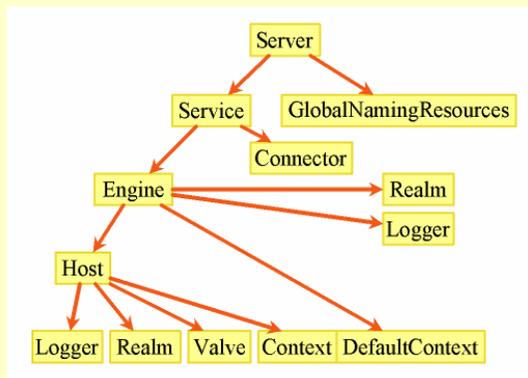


## 後半のまとめ： 外部構造（ネットワーク）の予測のための、半教師付予測法について紹介しました

- ネットワーク構造の予測問題は、2つのノードの関係の予測問題として捉えられる
- リンクのタイプが複数あるような場合を考慮
- ノード間の類似度情報が与えられるときに、これを利用して予測
  - ノードペア間の類似度は、ノード間の類似度のクロネッカー積／和によって定義
- 半教師付き手法「ラベル伝播」を、ノードペアに対して適用

# 今日は、構造をもったデータの解析についてお話ししました

- 構造をもったデータを内部構造と、外部構造にわけてお話ししました
  - 内部構造については、木やグラフを扱うためのカーネル法について
  - 外部構造については、複数タイプのリンク予測法について



```

MSGVYKVSIGSILQKDVTSEGETAILISLGLMTKEEKVPPAKMAMVASA 50
KANSI IFVSEDGSLSFEAPKETGETSKPGEKKEEKKVEVGVKFPFSAKV 100
KELIEGKSLTLDQDKIQKVLVEEYVKNLPRTAETYKPKIEIKCFKGVDFS 150
ISSLLSSGTKILDAILYSTYKDSAEHNFIFDVKVLSPDFIDSKLLVNNIE 200
TGNRAIKA AFCLVYNQGLPSKTSEERPLSKFVRETIFREKDLKANELCE 250
YLSSADPSLFP SQVFLKISLENLPTEVSSRCKMSIAGNKAMRYALLAQKF 300
DKDEIPVPTENVPTTSSEYMCKKEKIEKAKKIVDVLCSLASDFQAQVKMH 350
PLSPERSRKNFTLQLTSAIVTSLSYKGR LDMRKAIEEKKIEAFKRDENI 400
FGRLNALGQPTFPVLTNADADFSELSVEAVKTAYGKK 437
    
```