論理と計算

第１１回
高次推論：発想推論

担当：尾崎　知伸
ozaki.tomonobu@nihon-u.ac.jp

講義予定　※一部変更（前倒し）になる可能性があります

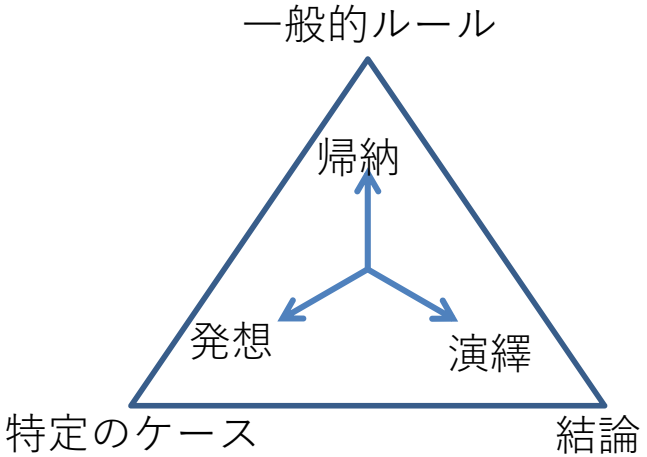
09/22	01. オリエンテーション と 論理を用いた問題解決の概要
09/29	02. 命題論理：構文・意味・解釈
10/06	03. 命題論理：推論
10/13	04. 命題論理：充足可能性問題
10/20	05. 命題論理：振り返りと演習（課題学習）
10/27	06. 述語論理：構文・意味・解釈
11/03	07. 述語論理：推論　※文化の日，文理学部授業日
11/10	08. 述語論理：論理プログラムの基礎
11/17	09. 述語論理：論理プログラムの発展
11/24	10. 述語論理：振り返りと演習（課題学習）
12/01	11. 高次推論：発想推論
12/08	12. 高次推論：帰納推論の基礎
12/15	13. 高次推論：帰納推論の発展
12/22	14. 高次推論：振り返りと演習（課題学習）
01/19	15. まとめと発展的話題

目次：今回の授業の内容

- 発想推論
  - 概要
  - 問題設定
- 発想論理プログラム
  - 拡張安定モデル
  - 解集合プログラミングによる拡張安定モデルの計算
- 応用例
  - 故障診断
  - バイオ
  - 演奏スキル

発想推論の概要

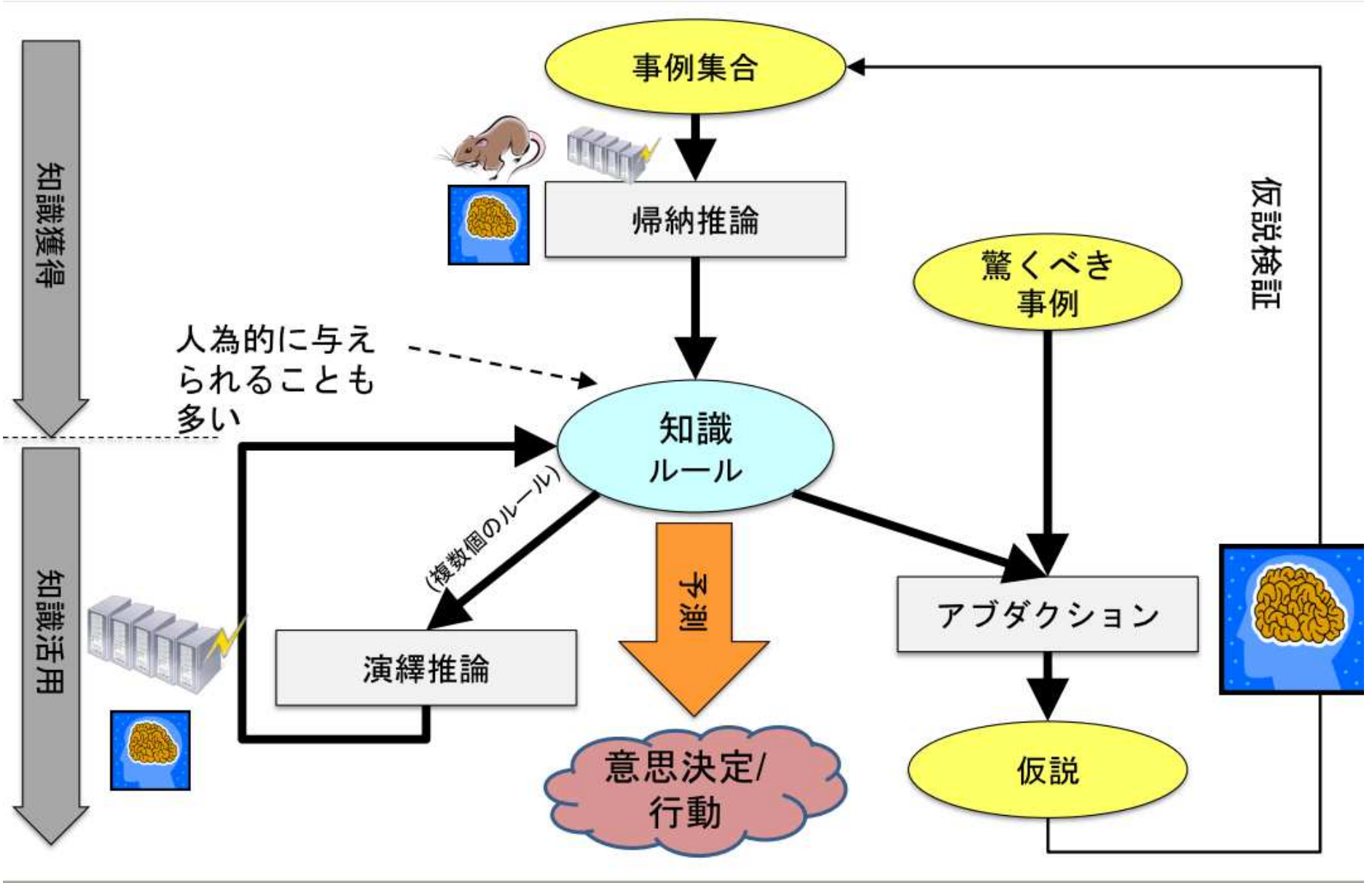
推論の種類



- 推論：知識をもとに，新しい結論を得ること
  - 既に分かっている情報・知識から，
    新しい情報・知識を導き出すこと
- 推論にはいくつかの種類がある
  - 演繹・発想・帰納・類推など
- 演繹推論（Deduction）
  - 一般的ルールを特定のケースに当てはめて結論を得る分析的過程
  - ルール「aならばb」とケース「a」から，結論「b」を導出
- 帰納推論（Induction）
  - 特定のケースと結論から，ルールを推論する合成的過程
  - ケース「a」と結論「b」の対から，ルール「a→b」を導出
  - いわゆる学習に相当
- 発想推論（Abduction）
  - 一般的ルールと結論化から，特定のケースを推論するもう一つの合成的過程
  - ルール「aならばb」と結論「b」から，ケース「a」を導出
  - ex. 病名診断．犯人捜し

常に正しい結果が得られる推論 vs 結果の正しさが保証されない推論

これまでの資料より



http://www.eccse.kobe-u.ac.jp/assets/images/distance_learning/life_science3/d84dedb94c7e352fbdc0c458b814a2ea59ecd46f.pdf より

発想推論の概要

- パースによる定義
  - 演繹推論：一般的ルールを適当のケースに当てはめて，結論を得る分析的過程
  - 発想推論：ルールと結論から，特定のケースを推論する合成的過程
  - 帰納推論：特定のケースと結論から，ルールを推論する合成的過程

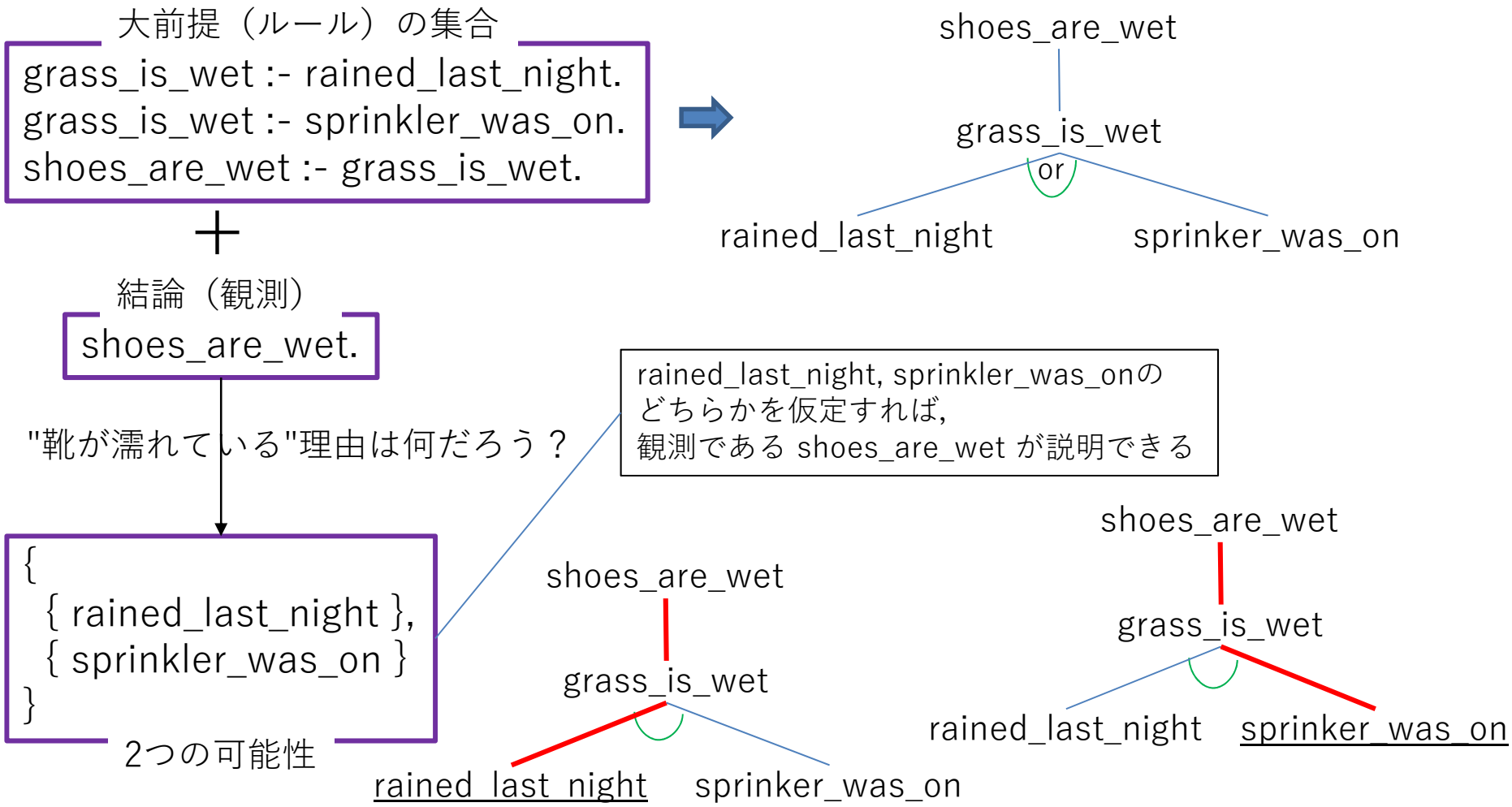
- 発想推論（abduction）
  - 「大前提 $\alpha \Rightarrow \beta$ および結論 β から小前提 α を導く」 $\dfrac{\beta, \alpha \Rightarrow \beta}{\alpha}$
  - 仮説推論・辻褄合わせの推論（状況を上手く説明できる根拠を導出する）
    - そう考えると，矛盾なく説明できる
    - 故障診断・医療診断・犯人捜し

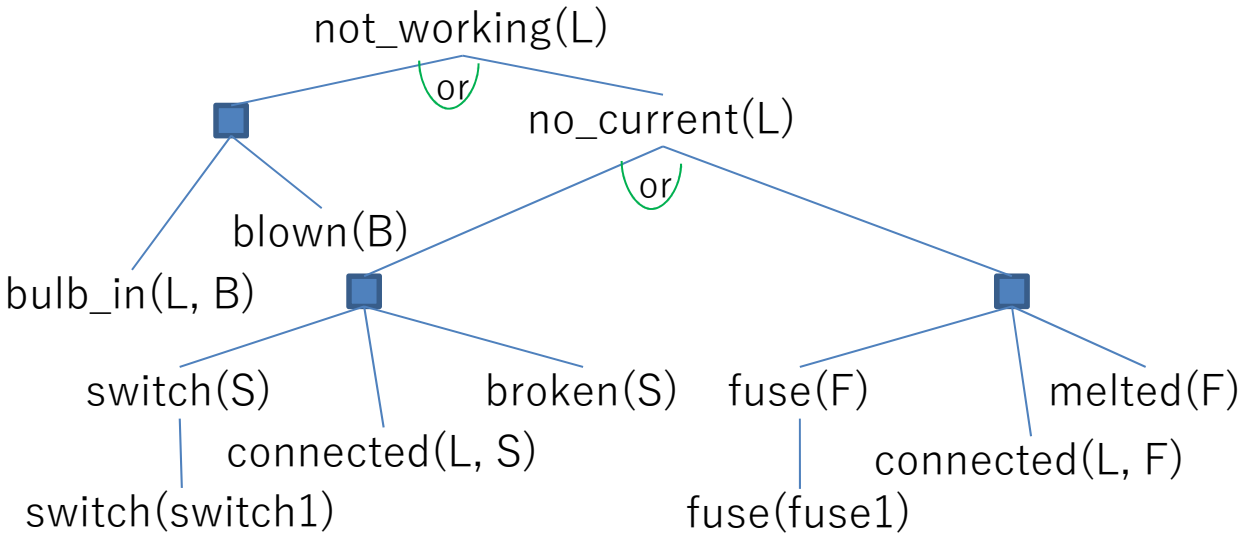
  - na：アリバイがない， c：犯人である，c⇒na：犯人にはアリバイがない
    - na と c⇒na から cを導く：アリバイが無くても犯人ではない
    - { c, c⇒na } |= na は成り立つ，{ na, c⇒na } |= c は成り立たない
      （必ずしも正しい推論ではない）

※必ずしも正しい推論ではない＝得られる結果が正しい（真実）とは限らない

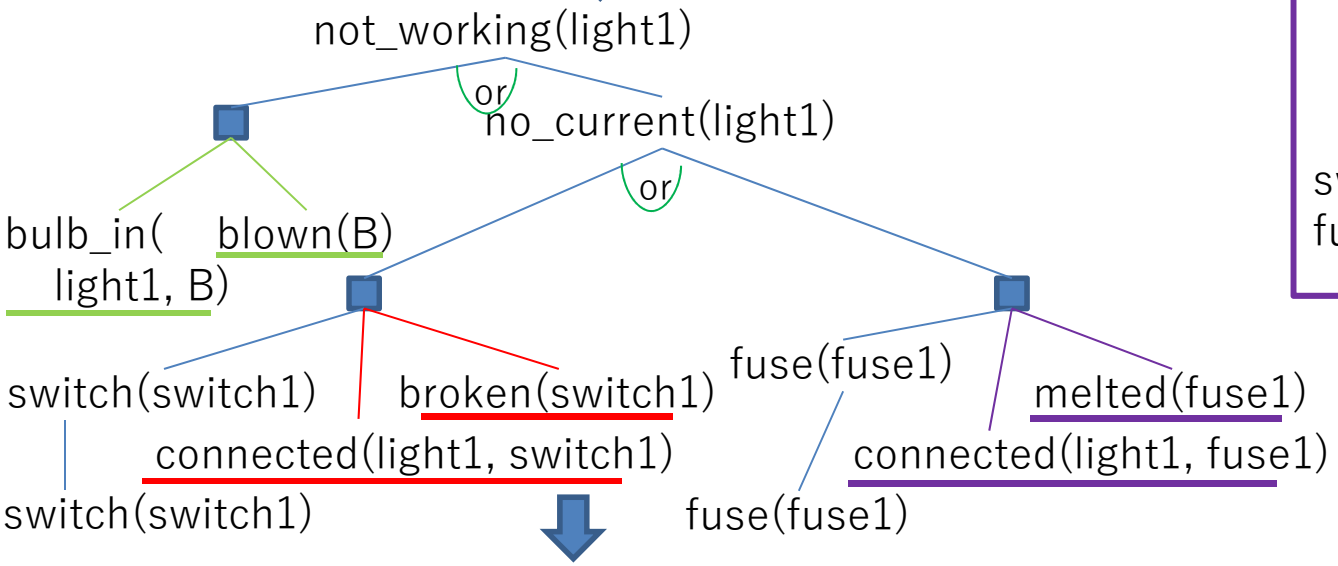
発想推論の例

- 「観測」と「モデル」に対して，辻褄が合う「説明」を求める推論
  - 発想推論における「説明」は，原子文（ファクト）の集合として獲得される





変数を定数に具体化



大前提（ルール）の集合

```
not_working( Light ):-
    bulb_in(Light, Bulb),
    blown( Bulb ).

not_working( Device ):-
    no_current( Device ).

no_current( Device ):-
    switch( Switch ),
    connected(Device, Switch),
    broken( Switch ).

no_current( Device ):-
    fuse( Fuse ),
    connected( Device, Fuse ),
    melted( Fuse ).
switch( switch1 ).
fuse( fuse1 ).
```

＋

結論（観測）

not_working(light1)

{ { connected(light1, switch1), broken(switch1) }, { connected(light1, fuse1), melted(fuse1) },
{ bulb_in(light1, B), blown(B) } }

※複数の事実を同時に仮定する必要がある場合もある

発想推論の形式的定義

- 発想推論（Abduction）
  - P $\not\models G$である論理式P と観測事実Gに対して，条件 P ∪ Δ ⊨ G かつ P ∪ Δは無矛盾 を満足する論理式（の集合）Δを求める
  - 論理式PにΔ を補うことによって，初めて観測事実Gが説明できる

- Δを発想的説明と呼ぶ．また，P ∪ Δを論理式Pの発想的拡張と呼ぶ
  - 発想的拡張は，通常の論理式となる点に注意

$$P = \left\{ \begin{array}{l} rained\_last\_night \Rightarrow grass\_is\_wet. \\ sprinkler\_was\_on \Rightarrow grass\_is\_wet. \\ grass\_is\_wet \Rightarrow shoes\_are\_wet. \end{array} \right\}$$

$$G = shoes\_are\_wet$$

$$\Delta_1 = \{rained\_last\_night\}$$

$$\Delta_2 = \{sprinkler\_was\_on\}$$

- 候補仮説集合（abducibles）：
  - Δ に含めることのできる文のクラス（利用者が与える）
  - 要は，利用者側で仮説の候補を準備するということ
  - Δに関して更なる条件を設けることが一般的
    - 候補仮説はすべて原子文に限る
    - 発想的説明は，与えられた一貫性制約を満足する
    - 発想的説明は，基本的（basic）でなければならない
      - grass_is_wetは，基本的ではない．∵さらに原因をさかのぼる必要がある
    - 発想的説明は，極小（minimal）でなければならない
      - Δ= { rained_last_night，sprinkler_was_on }は極小ではない

確認してみよう

- $P \nvDash G$であることを確認しよう
  - 真理値表を書いて確認しよう
- $P \cup \Delta_1 \vDash G$ であることを確認しよう
- $P \cup \Delta_2 \vDash G$ であることを確認しよう

$$P = \left\{ \begin{array}{l} rained\_last\_night \Rightarrow grass\_is\_wet. \\ sprinkler\_was\_on \Rightarrow grass\_is\_wet. \\ grass\_is\_wet \Rightarrow shoes\_are\_wet. \end{array} \right\}$$

$$G = shoes\_are\_wet$$

$$\Delta_1 = \{rained\_last\_night\}$$

$$\Delta_2 = \{sprinkler\_was\_on\}$$

発想論理プログラム

発想論理プログラム（abductive logic program）

- 発想論理プログラム：<P, A, IC>
  - 入力
  - P ：論理プログラム
  - G ：観測を表す論理プログラム（$P \not\models G$ ）
  - IC ：一貫性制約を表す論理プログラム
  - A ：候補仮説集合（基礎原子式）

  - 出力：以下の条件を満たす発想的説明（＝仮説）Δ （$\Delta \subseteq A$）
  - $P \cup \Delta \models G$ （発想的拡張はGを説明する）
  - $P \cup \Delta \models IC$ （発想的拡張は一貫性制約を満たす）
  - $P \cup \Delta$は無矛盾

- 発想論理プログラム
  - Pと共に観測Gを説明でき，かつ，一貫性制約ICに違反しない「ファクト集合」を見つける
    - ファクトアブダクション vs ルールアブダクション
  - PとIC
    - 意図・目的の観点から P と IC を区別している
    - P中に制約ICを書くことも可能
      - 論理式の観点からは，PとICの区別はない

発想論理プログラムのモデル

- モデル：プログラムを真にする基礎リテラルの集合
  - 発想推論の場合：元のプログラムP と 仮定した事実を用いて，観測を説明する
    → 仮定した事実を含めたモデルを考える
- 発想論理プログラムの意味論：拡張安定モデル（拡張解集合）
- 拡張安定モデル
  - 以下の条件を満たす基礎リテラルの集合$\mathrm{M}(\Delta)$を発想論理プログラム<P, A, IC >の拡張安定モデルと呼ぶ
    1. $\mathrm{M}(\Delta)$ は$\mathrm{P} \cup \Delta$の安定モデル（解集合）
    2. $\mathrm{M}(\Delta) \vDash \mathrm{IC}$
    3. $\Delta \subseteq A$
  - すなわち，Pの発想的拡張$\mathrm{P} \cup \Delta$の安定モデル$\mathrm{M}(\Delta)$のうち，
    一貫性制約を満たすものを「拡張安定モデル」と呼ぶ
  - この段階では，観測Gが（明示的には）考慮されていない．
    - 実際には，Gを含む拡張安定モデルを考える必要がある
      - 観測を「制約」と捉えると，すべてを統一的に扱うことができる．次ページ参照

解集合プログラムによる拡張安定集合の計算

- 発想論理プログラム<P, A, IC >と観測Gに対し，解集合プログラム

$$\mathrm{P} \cup IC \cup \{\leftarrow not\ G.\} \cup_{a \in A} \{a; not\ a \leftarrow.\}$$

を準備する

  - :- not G. の意味：Gがモデルに含まれないといけない
    - 明示的にGを考慮することになる
  - a; not a. の意味：aが成り立つか，成り立たないか
    - a をモデルに入れたら，上手く説明ができる？
    - a をモデルに入れなければ，上手く説明ができる？

  - Aの各部分集合とGを含む安定モデル（解集合）を考慮することになる

- 発想的説明Δの獲得
  - $\Delta = \mathrm{M}(\Delta) \cap A$
  - 得られた拡張安定モデル（＝解集合） $\mathrm{M}(\Delta)$とAとの積集合

確認してみよう

- 以下の各発想論理プログラムに対し，発想的説明を求めてみよう

$$P = \left\{ \begin{array}{l} p \leftarrow r, b, not\, q. \\ q \leftarrow a. \\ r. \end{array} \right\}, A = \{a, b\}, G = p$$

$$P = \left\{ \begin{array}{l} p; q \leftarrow not\, r. \\ r \leftarrow not\, a. \\ \neg q \leftarrow b. \end{array} \right\}, A = \{a, b\}, G = p$$

$$P = \left\{ \begin{array}{l} flies(X) \leftarrow bird(X), not\, abnormal(X). \\ abnormal(X) \leftarrow penguin(X). \\ bird(X) \leftarrow penguin(X). \\ bird(X) \leftarrow sparrow(X). \end{array} \right\},$$

$$A = \{penguin(tweety), sparrow(tweety)\}, G = flies(tweety)$$

発想的説明の選好

- 発想的論理プログラミングに対する発想的説明は複数考えられる
  - 発想的説明間に「順位」をつけたい
  - 参考：解集合でも同様・複数の解集合のどれが尤もらしい？
    - モデル間の順位と同様に，リテラルにも順位（種類）がある
    - 慎重な帰結（skeptical consequence） 「すべて」の解集合に含まれる基礎リテラル
    - 安易な帰結（credulous consequence）「ある」解集合に含まれる基礎リテラル

- 順位の付け方には，いろいろな方法が考えられる
  - 例１：仮定するリテラルの「数が少ない」方が良い→ 記述長最小原理など
  - 例２：仮定するリテラルが「成立確率が大きい」方が良い→ 確率論理プログラミングなど

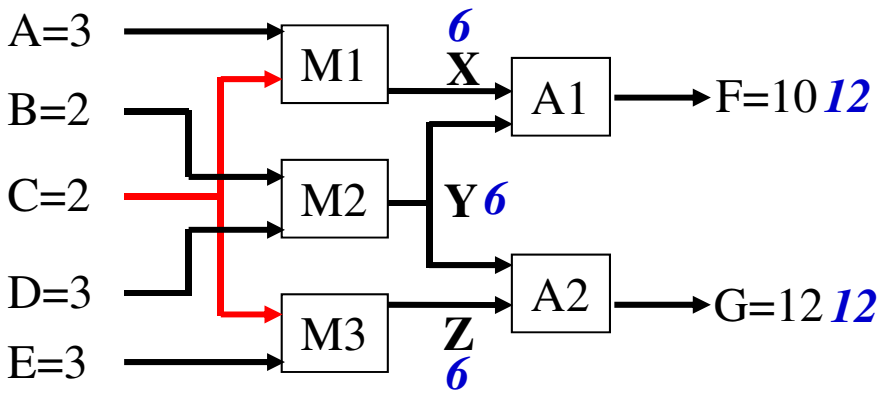
発想論理プログラムの応用

発想論理プログラムの応用の例

- プラニング：
  - M. Shanahan: An abductive event calculus planner, J. of Logic Programming, 44:(1-3), 207-240, 2000.
- 航空会社の乗務員のスケジューリング：
  - M.Denecker他：Abduction in Logic Programming, Computational Logic: Logic Programming and Beyond, LNCS2407, pp.99-134,2002.
- 音楽：
  - I. Kobayashi他: Modeling Physical Skill Discovery and Diagnosis by Abduction, 人工知能学会論文誌， Vol.23, No.3, pp.127-140, 2008.
- 法的推論 ：
  - 佐藤 健：事例ベース推論における動的類似性の仮説論理プログラミングによる実現， 人工知能学会誌Vol. 12, No.6, pp. 901–910, 1997.
- ソフトウェア工学：
  - 佐藤 健：仮説論理プログラミングによる極小変更仕様の計算およびその応用. コンピュータソフトウェア別冊, 「ソフトウェア発展」, 日本ソフトウェア科学会編, pp. 109–121, 2000.
- エージェント：
  - 佐藤 健 他：エージェント間通信におけるアブダクションによる投機的計算. コンピュータソフトウェア, Vol. 20 No.1, pp. 27–35, 2003.
- ゲーム：
  - 上原 貴夫：コンピュータブリッジにおけるアブダクションの応用， 電子通信情報学会論文誌D-II, Vol. J77-D-II, No.11, p.2255-2264, 1994.
- 医学：
  - O. Ray 他：Abductive Logic Programming in the Clinical Management of HIV/AIDS, ECAI'06, pp. 437-441, 2006.
- 遺伝子ネットワーク：
  - I. Papatheodorou 他： Inference of gene relations from microarray data by abduction, LPNMR'05, pp.389-393, 2005

故障診断：無矛盾性に基づく診断

- 観測とモデル，入力パラメタに矛盾しない仮説を導出する
  - 「正常」と仮定する部品集合を極大化する
- 入力
  - システムのモデル（SD）
  - 入力パラメタ（IN）
  - 観測（OUT）
  - 各部品のモード（COMPS）：正常 or 異常のいずれか → 命題変数
- 出力
  - 条件「SD∪IN ∪HYP ∪OUTは充足可能」を満たすHYP⊆COMPS
- 特徴
  - 正常時のモデルを利用＝故障時のモデルは不要
  - 故障部品の同定までが可能．どの様な不具合化は判別はできない

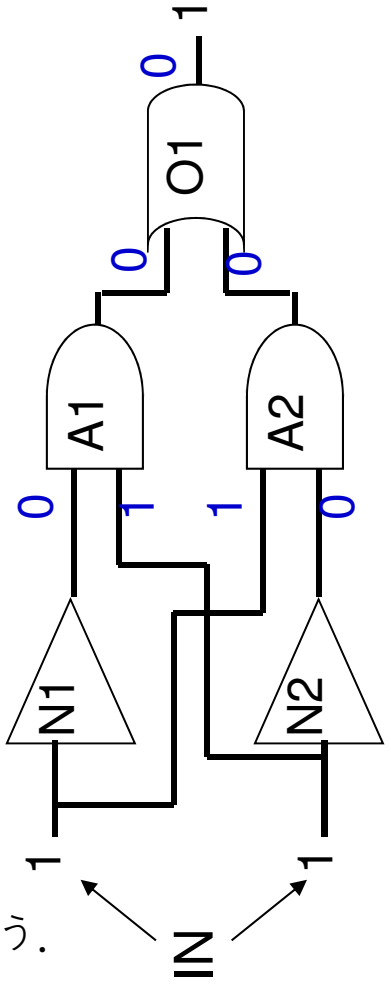


仮説（壊れた部品の集合）{
{A1},
{M1},
{A2,M2},
{M3,M2}
}

故障診断：アブダクションに基づく診断

- 入力パラメタ・モデルと共に観測を説明する仮説を導出する
  - 異常と仮定する部品とそのモードを出力する

- 入力
  - システムのモデル（SD）
  - 入力パラメタ（IN）
  - 観測（OUT）
  - 各部品のモード（COMPS）：正常時 / 異常時における挙動
- 出力
  - 条件「SD∪IN ∪HYP|=OUT」を満たすHYP⊆COMPS
- 特徴
  - 正常時+故障時のモデルを利用
  - 故障部品と故障モードの同定

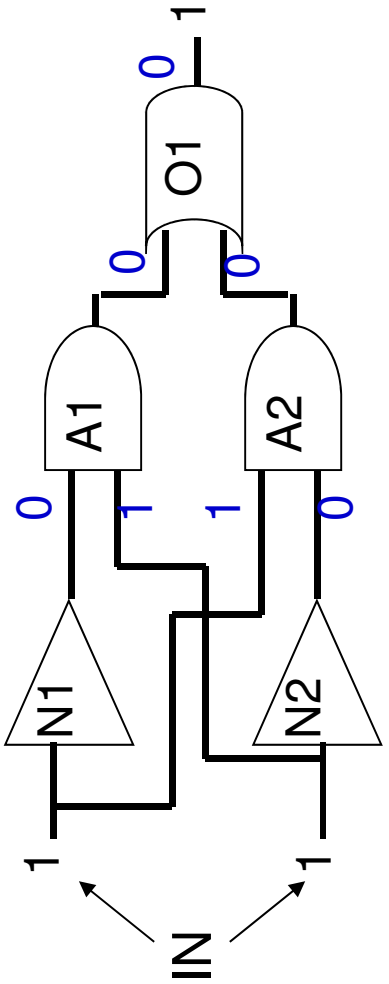


N1, N2の故障モード：ショート（入力をそのまま出力してしまう）
A1, A2, O1の故障モード：反転(本来の出力結果と逆の結果を出力してしまう.
仮説：{ {N1がショート}, {N2がショート},
{A1が反転}, {A2が反転},
{O1が反転} }

circuit.lp

```
out(Out):- in(in1, In1), notGate(n1, In1, NOut1),
           in(in2, In2), notGate(n2, In2, NOut2),
           andGate(a1, NOut1, In2, AOut1),
           andGate(a2, NOut2, In1, AOut2),
           orGate(o1, AOut1, AOut2, Out).

gate(o1, or).
orGate(N, In1, In2, Out1):- gate(N,or), normal(N), or_(In1, In2, Out1).
orGate(N, In1, In2, Out1):- gate(N,or),   rev(N), or_(In1, In2, O), not_(O, Out1).
or_(0,0,0).   or_(0,1,1).   or_(1,0,1).   or_(1,1,1).

gate(a1, and).  gate(a2,and).
andGate(N, In1, In2, Out1):- gate(N,and), normal(N), and_(In1, In2, Out1).
andGate(N, In1, In2, Out1):- gate(N,and),   rev(N), and_(In1, In2, O), not_(O, Out1).
and_(0,0,0).   and_(0,1,0).   and_(1,0,0).   and_(1,1,1).

gate(n1, notg).  gate(n2,notg).
notGate(N, In1, Out1):- gate(N,notg), normal(N), not_(In1, Out1).
notGate(N, In1,  In1):- gate(N,notg),  short(N), not_(In1, _).
not_(0,1).   not_(1,0).

%abducibles
1{normal(X); rev(X)}1 :- gate(X, and).
1{normal(X); rev(X)}1 :- gate(X, or).
1{normal(X);short(X)}1:- gate(X, notg).

% for single fault
abnormal(X):- rev(X).
abnormal(X):- short(X).
{abnormal(X):gate(X,_)} 1.
:- normal(X), abnormal(X).
```

```
%input and observation
:- not out(1).
in( in1, 1 ).
in( in2, 1 ).
```

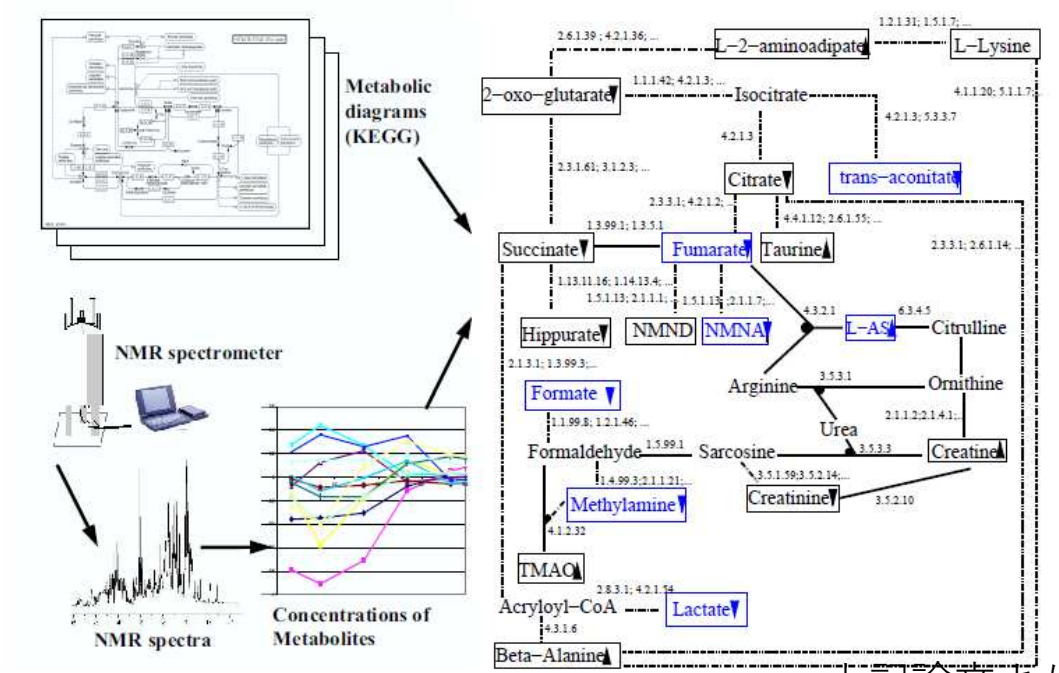


バイオインフォマティクス

- A. Tamadoni-Nezhad 他：Abduction and Induction for modeling inhibition in metabolic networks, Int. W.S. on the integration of abduction and induction in AI, 2005.

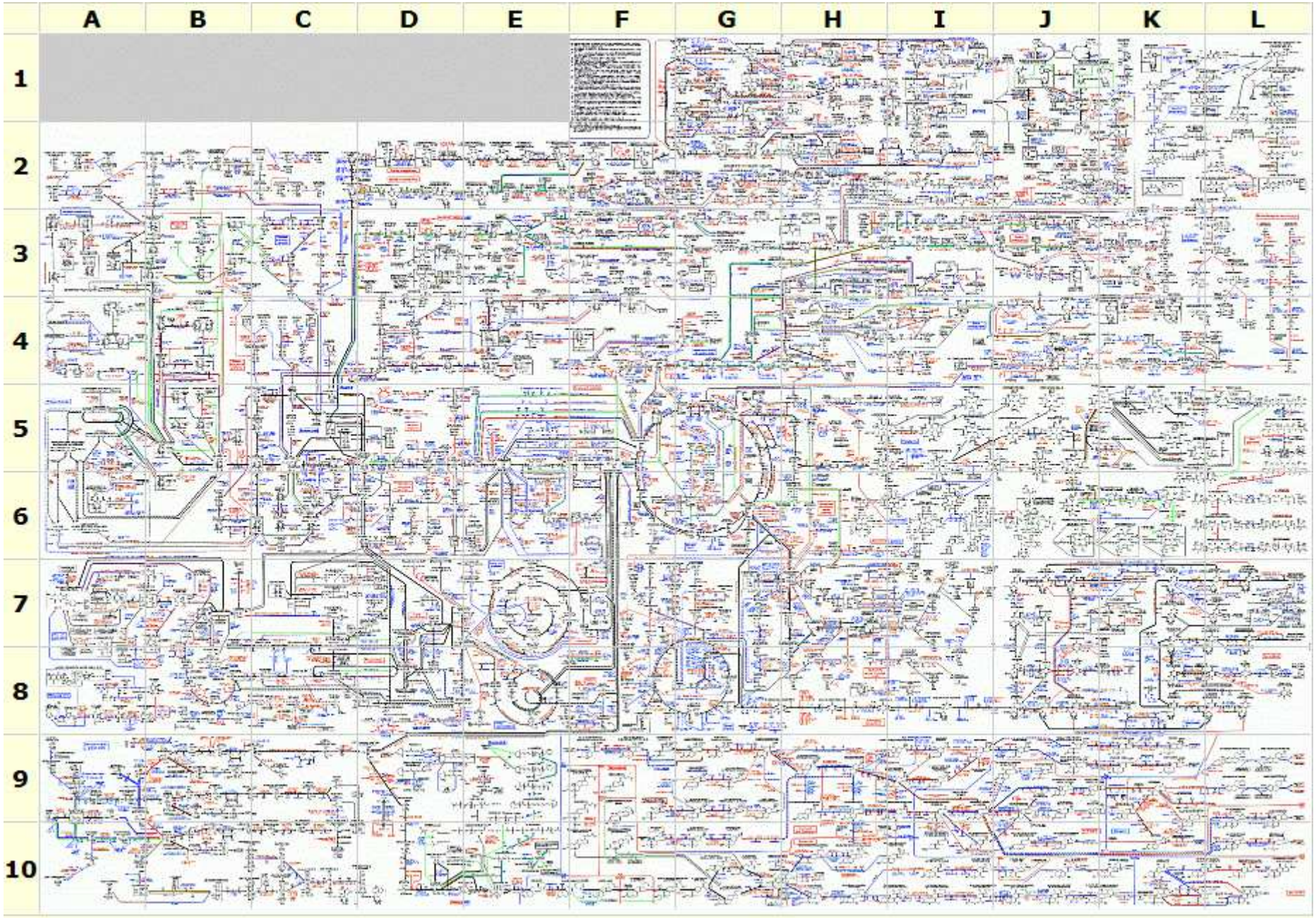
- 代謝ネットワーク（Metabolic Network ）：
  - 生体内の代謝産物の反応ネットワーク．酵素によって化合物を別の化合物へと変換する一連の化学反応のネットワーク．

- 代謝産物の濃度に関する，毒性物質の影響を分析するためのモデル化
  - 毒性物により，代謝ネットワークのどの部分の反応が抑制されたのかを推論する



上記論文より

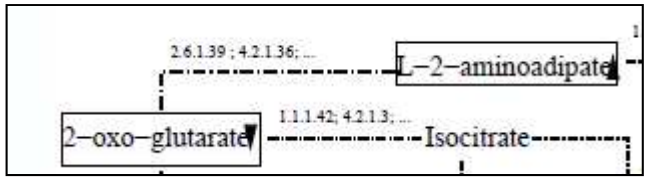
Biochemical Pathways - Metabolic Pathways



http://web.expasy.org/cgi-bin/pathways/show_thumbnails.pl

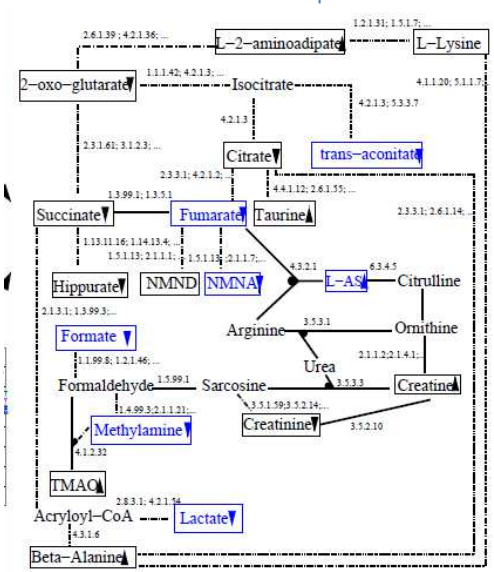
本来は，ある"毒素"の下での濃度ということで，
毒素に対応する項が必要だが，
毒素毎にモデル化を行うことで，これを回避

- 観測：
  - concentration(Metabolite, Level, Time).
  - 時刻Timeにおいて，代謝物Metaboliteの濃度はLevelである．
    - Level ∈ {up, down} … 濃度が上がる／下がる



- 関連知識（ネットワーク構造）：
  - reaction_node(Metabolites1, Enzymes, Metabolites2).
    - reaction_node('l2aminoadipate', '2.6.1.39', '2oxoglutarate').
  - 代謝物Metabolites1とMetabolites2は，酵素Enzymesにより反応する

- 仮定可能述語（Abducible 述語）：
  - inhibited(Enzyme, Metabolites1, Metabolites2, Time).
    - inhibited('2.6.1.39', 'l2amino..', '2oxoglutarate', 8).
  - 時刻Timeにおいて，Enzymeによる
    Metabolites1, Metabolites2間の反応が抑制されている



- 関連知識2（濃度の変化のモデル）
  - concentration(X, down, T):- reactionnode(X,Enz,Y), inhibited(Enz, Y, X, T).
  - concentration(X, down, T):- reactionnode(X,Enz,Y),
    not inhibited(Enz,Y,X,T), concentration(Y,down,T).
  - concentration(X, up, T):- reactionnode(Y,Enz,X),inhibited(Enz,X,Y,T).
- 一貫性制約
  - :- concentration(X, down, T), concentration(X, up, T).

- 実験では，Abduction + Inductionを行っている.
  - CWA　（閉世界仮説に基づき）inhibited/5, concentration/3を準備
    - inhibited(Enz, false, X,Y,T):- reactionnode(Y,Enz,X), not(inhibited(Enz, true, _, _)).
    - concentration1(X, up, T):- concentartion(X, up,T), not(concentration(X,down,T)).
    - concentration1(X, down, T).
- 結果
  - inhibited('2.6.1.39',true,'l2aminoadipate','2oxoglutarate',8).
  - inhibited('2.3.1.61',false,'2oxoglutarate','succinate',8).
  - inhibited('1.13.11.16',false,'succinate','hippurate',8).
  - inhibited('2.6.1.-',true,'taurine','citrate',8). などのアトム
  - これらを一般化し．．
  - inhibited(Enz, true, M1, M2, T):- reactionnode(M2, Enz, M1), class(Enz, 'aminotransferase').

アミノ基転移酵素による反応は，
毒素により抑制される

楽器演奏の身体スキル

- 古川康一他，数理論理学，8章より
  - より詳しくは，古川康一他，身体スキル発想支援の非単調推論によるモデル化について，SIG-FPAI-A603-03,pp.13-19, 2007.

- チェロの演奏スキルに関する推論
  - 高速ポジション移動を行う奏法を求める問題
  - 身体と動作の関係をモデル化

- チェロの演奏方法：　右手で弓を持つ．左手で弦を抑える．
- ポジション移動：　左手の位置を（平行）移動させること

- ポジション移動の仕方には，2つの方法が考えられる．問題は，これらが正しいか？
  - 肩関節 及び 肘関節の開閉（内外転，adduction/abduction）によって，肘を上下に動かす方法
  - 体温計を振るように，肘を中心に前腕を振って（上腕の内外転，incycloductin/excycloduction)手の位置を上下に動かす方法

- 制約：慣性モーメントが大きいと，高速な動きはできない

```
%ポジションシフトの2方法
rapid_position_shift :- rapid_move,
        add_and_abduction_of_shoulder,
        add_and_abduction_of_elbow.
rapid_position_shift :- rapid_move,
        in_and_excycloduction_of_upper_arm,
        add_and_abduction_of_elbow.

% 肩関節の内外転は，大きな慣性モーメントの発生
big_inertia_moment :-
add_and_abduction_of_shoulder.

%一貫性制約 慣性モーメントが大きいと高速移動は不可能
:- rapid_move, big_inertia_moment.
```

{ rapid_move,　in_and_excycloduction_of_upper_arm,
　add_and_abduction_of_elbow }

すなわち，第二の方法だけが，奏法として正しいことが分かる．