4,関係が深い表現推論法

4.1　制約充足問題(CSP:Constraint Satisfaction Problem)

変数：X1,X2,X3

変数の領域：Di(Xi のとりうる離散値)

制約：Cij(Xi,Xj 間の可能な組み合わせ)

Ex)

変数　X1,X2,X3

D1={a,b} D2={e,f} D3={c,d,g}

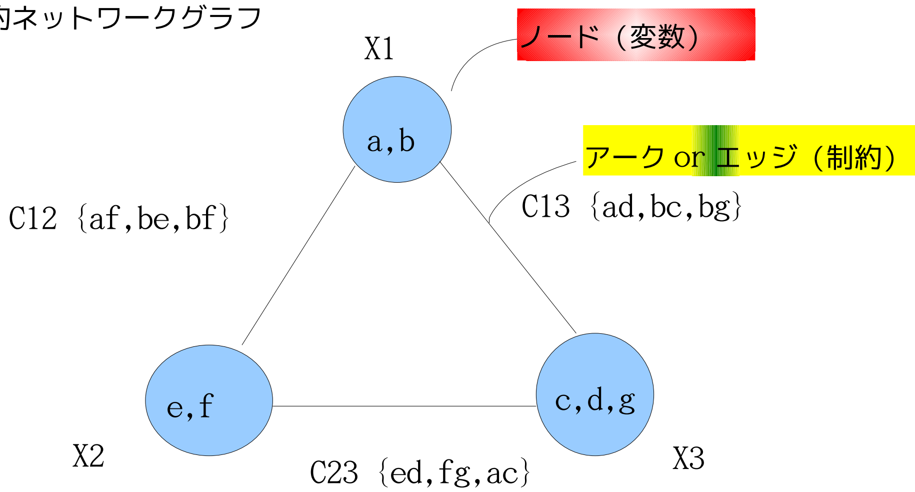
制約　C12={af,be,bf}

C13={ad,bc,bg}

C23={ed,fg,ac}

２項制約←一般にはｎ項制約（２項制約に変換可能）

制約ネットワークグラフ



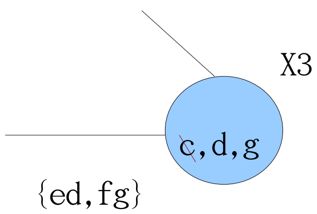
この場合の解

{X1,X2,X3} = {b,f,g}　　NP 完全問題　３SAT

(1)局所整合アルゴリズム (Local Consistency Algorithm)

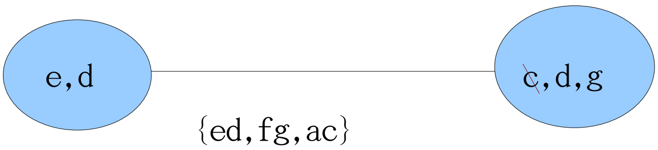
1、ノード整合アルゴリズム

１ノードに関して不整合値を除く



2、アーク整合アルゴリズム（Waltzのフィルタリング）

２ノード間の２項制約に関して不可能値を除く



繰り返す計算量　O(e*d^2)　e:アーク数　d:各変数の領域のサイズ

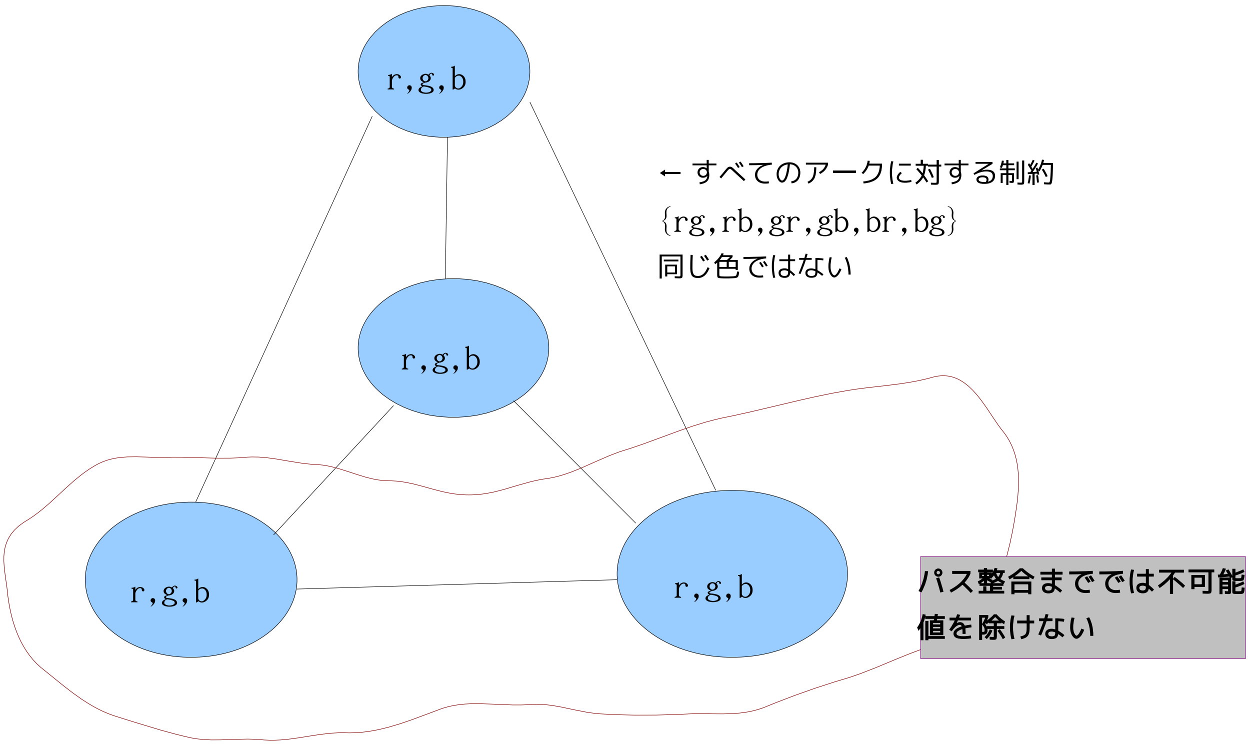
3、パス整合アルゴリズム

２ノード間の他ノード経由すべてのパスについての不可能値を除く

X3=d とすると、X1=a，X2=e でなければばならない。

ところがC12={ae}は存在しない。

パス整合まででは矛盾の組が残る例



4、k整合アルゴリズム　　（k≧4 →　計算時間が増大　→　？）

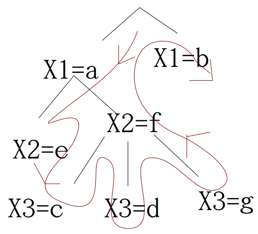
k ノード間での不整合値を除く

k=ノード数（変数の数）まで行うと、元のCSPをといたことになる。

(2)木探索

深さ優先 ―― 1ノードずつ値を決めていき、可能値がなくなったらバックトラック
幅優先

基本だが、効率よくない



(3)Lookahead

制約の受動的使用

生成した値を制約によりチェック

制約の能動的利用

値の生成に制約を利用

深さ優先探索と組み合わせて利用

1、フォワードチェッキング(forward checking)　計算量小

具体化した値と具体化していない変数間の制約より、
可能な値の候補から矛盾するものを排除
値の候補がなくなった変数があるとき→早期のバックトラック

2、Full Lookahead

具体化してない変数 Xi の可能な値{vi1,vi2,…, vil}について
他の具体化してない変数 Xj の可能な値{vj1,vj2,…,vjm}と制約を評価。
vik が vj1 ～ vjm　のすべてと矛盾するなら、vik を候補から外す
各 vjk についてすべての j について行い、Xi の値の候補が無くなればバックトラック

3、Partial Lookahead

j として i 以降のものだけについて行う。
Full よりも幾分計算量は少なくなるが、まだ大きい

Ex)８クイーン：クイーン（飛車＋角）が互いを取れない位置になるように８つ並べる

	X1	X2	X3	X4	X5	X6	X7	X8
1	○	×	×	×	×	×	×	×
2		×	×	×	×	×	c	×
3		○	×	×	×	×	×	×
4			×	×	×	a	×	×
5			◎	×	×	×	×	×
6				×	×	×	×	×
7				×	×	×	×	b
8						×	×	×

x:フォワードチェッキングによる値の排除

◎を決めたとき、X6=4 （a） しか値を取れないので、これと矛盾する x を候補から除く

X8=7（b）しか残らず、×を除く

X7=2（c） しか残らず、×を除く

X4=8,X5=8 しか残らず、結局失敗。

→◎の選択が解にならないので、バックトラック

１つの解

		△		○			
○△							
			○			△	
				△	○		
							○△
	○△						
			△			○	
		○			△		

左右上下逆　９０°回転も解

８クイーン、９４個の解

CSP で良いとされているシステマティックな解法

アーク整合アルゴリズムによる前処理、

フォワードチェッキング付き深さ優先木探索

単解を求める確率的解法（特に解が空間に分散して存在するとき）例：n クイーン問題

山登り＋ランダム再スタート

変数具体化順序のヒューリスティックス

可能値が少ない変数から具体化