The Constraint Satisfaction Dichotomy Theorem for ~~Dummies~~ Beginners

Tutorial – Part 1

Ross Willard

University of Waterloo

BLAST 2019
CU Boulder, May 20, 2019

BLAST 2010 (Boulder)

- I gave a tutorial on the **Constraint Satisfaction Problem Dichotomy Conjecture** and its connection to universal algebra.

News flash: the conjecture is now a theorem!

So I'm back to give a post-mortem.

Tutorial Outline

Today: recall the problem and connection to universal algebra.

Wednesday: fool around with linear equations.

Friday: Formulate a key lemma in one of the proofs of the theorem.

Constraint Satisfaction Problems (CSP)

Fix $\mathbf{A} = (A, \Gamma)$, a finite relational structure in a finite signature.

Definition

A CSP instance over $\mathbf{A}$ is a conjunction $\bigwedge_{t \in T} C_t$ of pure atomic formulas in the signature of $\mathbf{A}$. ("pure" means "no equalities")

Definition

A CSP instance over $\mathbf{A}$ is consistent (or is satisfiable, or has a solution) if the relation it defines in $\mathbf{A}$ is nonempty.

Definition

CSP($\mathbf{A}$) is the decision problem which, given a CSP instance over $\mathbf{A}$, asks whether the instance is consistent (in $\mathbf{A}$).

Example 1

$\mathbf{2}_{\leq} := (\{0,1\}, \leq, \{0\}, \{1\})$.

Example of a CSP instance over $\mathbf{2}_{\leq}$:

$$(x_0 = 1) \wedge (x_0 \leq x_1) \wedge (x_1 \leq x_2) \wedge \cdots \wedge (x_{n-1} \leq x_n) \wedge (x_n = 0)$$

Quiz: Is this instance consistent, or inconsistent? Inconsistent.

In general, an instance over $\mathbf{2}_{\leq}$ is inconsistent iff it contains a subformula like the one above. This condition is easy to test. Hence:

CSP($\mathbf{2}_{\leq}$) is in P.

Example 2

Fix a prime p.

$\mathbf{Z}_p^{3lin}$ is the following structure:

- For $b, c, d \in \mathbb{Z}_p$, let L_{bcd} be the set of solutions in $\mathbb{Z}_p$ to

$$x + by + cz \;=\; d.$$

- $\mathbf{Z}_p^{3lin} := (\mathbb{Z}_p,\ \{L_{bcd}\}_{b,c,d \in \mathbb{Z}_p})$.

Instances of CSP($\mathbf{Z}_p^{3lin}$) "are" systems of 3-variable linear equations over $\mathbb{Z}_p$.

Inconsistencies can't quite be seen "by inspection," but they can be discovered in polynomial time, e.g. by Gaussian elimination. Hence:

CSP($\mathbf{Z}_p^{3lin}$) is in P (for each p).

Not every CSP($\mathbf{A}$) is easy

Consider

$$\mathbf{2}_{3sat} := (\{0,1\},\ R_{000},\ R_{001},\ R_{011},\ R_{111})$$

where $R_{abc} = \{0,1\}^3 \setminus \{(a,b,c))\}$.

E.g., $R_{001}(x,y,z)$ encodes "$x = 1$ or $y = 1$ or $z = 0$"

Instances of CSP($\mathbf{2}_{3sat}$) encode 3SAT and vice versa.

Thus CSP($\mathbf{2}_{3sat}$) is NP-complete.

Dichotomy Conjecture

If P $\neq$ NP, then there exist problems in NP which are neither in P nor NP-complete. (Ladner, 1975)

CSP Dichotomy Question (T. Feder and M. Vardi, STOC 1993)

Is it true that for every finite structure $\mathbf{A}$ in a finite signature, CSP($\mathbf{A}$) is NP-complete or in P?

CSP Dichotomy Conjecture

Yes.

Since around 1998, interest in this conjecture exploded, migrated to universal algebra, and became the defining problem in universal algebra for the last 15 years.

Two happy guys

Andrei Bulatov Dmitriy Zhuk

Update: the Conjecture is proved

Andrei Bulatov and Dmitriy Zhuk, independently, have announced proofs of the Dichotomy Conjecture (FOCS 2017).

Full proofs are on the arXiv. (101 and 47 pages)

In Dec 2017, Lance Fortnow wrote in his *Computational Complexity* blog `https://blog.computationalcomplexity.org/2017/`

> *"The theorem of the year goes to the resolution of the dichotomy conjecture" (by Bulatov and Zhuk).*

THE END

An easy reduction

I will say a structure $\mathbf{A} = (A, \Gamma)$ has constants if for each $a \in A$ there exists $R \in \Gamma$ such that for all $x \in A$,

$$x = a \iff (x, x, \ldots, x) \in R.$$

Each of the examples named earlier has constants.

$\mathbf{2}_{\leq} = (\{0,1\},\ \leq,\ \{0\},\ \{1\})$:

$$x = a \iff x \in \{a\}.$$

$\mathbf{Z}_p^{3lin} = (\mathbb{Z}_p,\ \{L_{bcd} : b, c, d \in \mathbb{Z}_p\})$:

$$x = a \iff (x, x, x) \in L_{00a}.$$

$\mathbf{2}_{3sat} = (\{0,1\},\ R_{000},\ R_{001},\ R_{011},\ R_{111})$:

$$x = 0 \iff (x, x, x) \in R_{111}; \quad \text{similarly for 1.}$$

An easy reduction

I will say a structure $\mathbf{A} = (A, \Gamma)$ has constants if for each $a \in A$ there exists $R \in \Gamma$ such that for all $x \in A$,

$$x = a \iff (x, x, \ldots, x) \in R.$$

Each of the examples named earlier has constants.

$\mathbf{2}_{\leq} = (\{0,1\},\ \leq,\ \{0\},\ \{1\})$:

$$x = a \iff x \in \{a\}.$$

Lemma. (Feder, Vardi)

To prove the Dichotomy Conjecture, it is enough to prove it for structures with constants.

Compatible algebra

Definition

An algebra is a pair $\mathbb{A} = (A, \mathcal{F})$ where A is a set and $\mathcal{F}$ is a family of operations on A.

Definition

A subuniverse of an algebra $\mathbb{A} = (A, \mathcal{F})$ is a subset $B \subseteq A$ which is closed under the operations in $\mathcal{F}$.

Notation: $B \leq \mathbb{A}$ and also $\mathbb{B} \leq \mathbb{A}$.

Definition

Let $\mathbf{A} = (A, \Gamma)$ be a relational structure and $\mathbb{A} = (A, \mathcal{F})$ an algebra with the same universe.

$\mathbf{A}$ is compatible with $\mathbb{A}$ if each $R \in \Gamma$ (n-ary) is a subuniverse of $\mathbb{A}^n$.

Examples

$\mathbf{2}_{\leq} = (\{0,1\}, \leq, \{0\}, \{1\})$ is compatible with $\mathbb{2}_{lat} = (\{0,1\}, \wedge, \vee)$.

Proof.

That is, $\{0\}$ and $\{1\}$ are subuniverses of $\mathbb{2}_{lat}$ (obvious),

and $\leq$ (i.e., $\{(0,0),(0,1),(1,1)\}$) is a subuniverse of $(\mathbb{2}_{lat})^2$. □

$\mathbf{Z}_p^{3lin} = (\mathbb{Z}_p, \{L_{bcd}\}_{b,c,d})$ is compatible with $\mathbb{Z}_p^{aff} = (\mathbb{Z}_p, x-y+z)$.

Proof.

Suppose $\mathbf{x}, \mathbf{y}, \mathbf{z} \in L_{bcd}$ where $\mathbf{x} = (x_1, x_2, x_3)$ etc.

Must check that $\mathbf{x} - \mathbf{y} + \mathbf{z} \in L_{bcd}$. It's easy. □

Examples (continued)

$\mathbf{2}_{3sat} = (\{0,1\}, R_{000}, R_{001}, R_{011}, R_{111})$ is not compatible with any interesting algebra. More precisely:

Fact

$\mathbf{2}_{3sat}$ is compatible with $(\{0,1\}, \mathcal{F}) \Leftrightarrow$ each operation in $\mathcal{F}$ is a projection.

Intuition: CSP($\mathbf{A}$) is easier when $\mathbf{A}$ is compatible with a "nontrivial" algebra $\mathbb{A}$, and is harder when it isn't.

Problem: What notion of "nontriviality" should be the dividing line between CSP($\mathbf{A}$) being easy (in P) and hard (NP-complete)?

Recall: we only need to consider structures having constants.

Easy Fact: If **A** has constants, any algebra $\mathbb{A}$ to which **A** is compatible must be idempotent (each operation f satisfies $(\forall x)\ f(x, x, \ldots, x) = x$).

$\exists$ a classical notion of idempotent nontriviality from universal algebra:

Definition (cf. W. Taylor, 1977)

An idempotent algebra $\mathbb{A}$ is Taylor if it has a term operation $t(x_1, \ldots, x_n)$ $(n > 1)$ such that for each position i, $\mathbb{A}$ satisfies an identity of the form

$$(\forall x, y, \ldots)\quad t(\text{vars}, x, \text{vars}') = t(\text{vars}'', y, \text{vars}'''),$$

with the first x in position i and the second y in position i.

Intuition: t is forced by identities to not be a projection.

Idempotent algebra $\mathbb{A}$ is Taylor if it has a term $t(x_1, \ldots, x_n)$ such that $\forall i$, $\mathbb{A}$ satisfies an identity of the form $t(\ldots, x, \ldots) = t(\ldots, y, \ldots)$.

Examples

$\mathfrak{2}_{lat} = (\{0,1\}, \wedge, \vee)$ is Taylor.

Proof: let $t(x,y) = x \wedge y$. $\quad \mathfrak{2}_{lat} \models t(x,y) = t(y,x)$. □

$\mathbb{Z}_p^{aff} = (\mathbb{Z}_p,\ x-y+z)$ is Taylor.

Proof: let $t(x,y,z) = x - y + z$.

$$\mathbb{Z}_p^{aff} \models t(x,x,y) = t(y,y,y)\ \&\ t(x,y,y) = t(x,x,x).$$ □

Clearly $(\{0,1\}, \{\text{projections}\})$ is not Taylor.

CSP Algebraic Dichotomy Conjecture

Algebraic Dichotomy Conjecture (Bulatov, Jeavons, Krokhin 2005)

Let $\mathbf{A}$ be a finite relational structure in a finite signature. Assume $\mathbf{A}$ has constants.

1. If $\mathbf{A}$ is compatible with some Taylor algebra, then CSP($\mathbf{A}$) is in P.
2. Otherwise, CSP($\mathbf{A}$) is NP-complete.

Algebraic Dichotomy Theorem (Bulatov 2017, Zhuk 2017)

The ADC is true.

Proof sketch.

Just need to prove (1). □

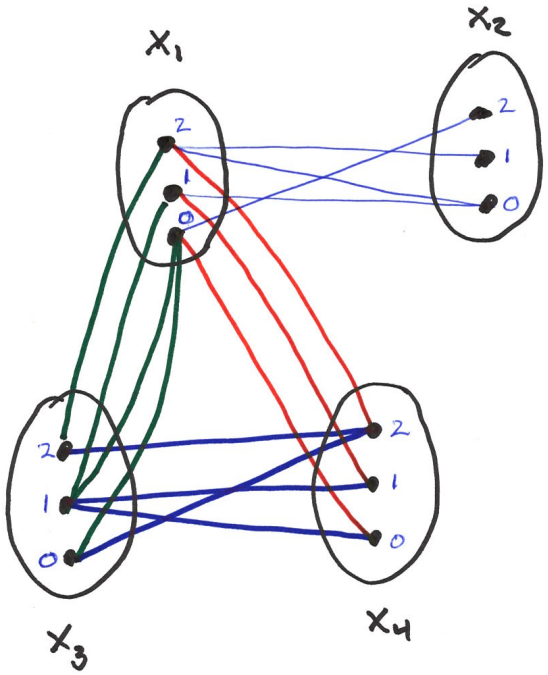
Preparation for next lectures – preprocessing

Suppose $\mathbf{A} = (A, \Gamma)$ is compatible with $\mathbb{A} = (A, \mathcal{F})$.

Let $\Theta = \bigwedge_{t=1}^{m} C_t$ be a CSP instance over $\mathbf{A}$.

Let $\mathrm{Var}(\Theta) = \{x_1, \ldots, x_n\}$.

We can visualize Θ by drawing a copy of A for each variable, and for each constraint $C_t = R(x_{i_1}, \ldots, x_{i_k})$, adding the tuples in R as hyper-edges between the copies of A corresponding to $x_{i_1}, \ldots, x_{i-1}$.



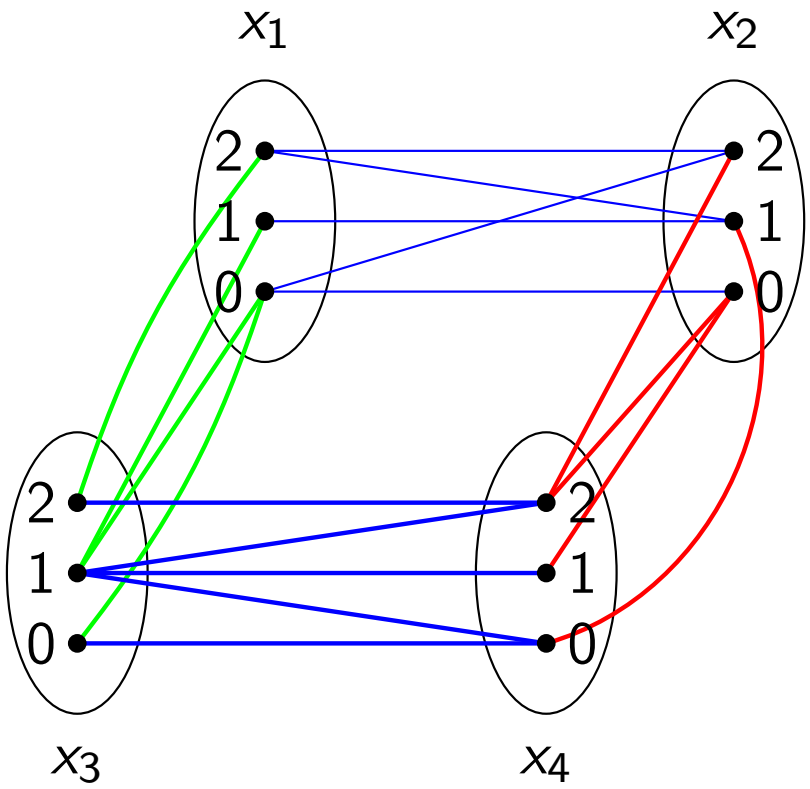
This is the picture of $R_1(x_1, x_2) \wedge R_2(x_1, x_3) \wedge R_3(x_1, x_4) \wedge R_4(x_3, x_4)$.

The copy of A corresponding to the variable x_i is called the **domain** for x_i (by computer scientists) or the **potato** for x_i (by universal algebraists).

This picture is called the **microstructure graph** of Θ (by computer scientists) or the **potato diagram** (by universal algebraists) .

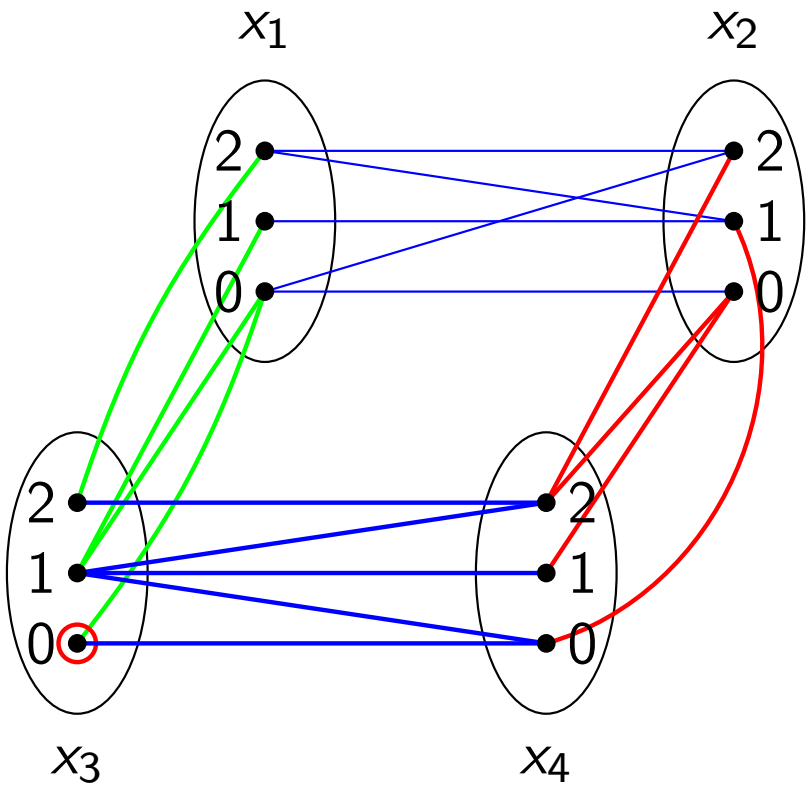
A solution to Θ is simply a choice of values in each potato satisfying every constraint.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



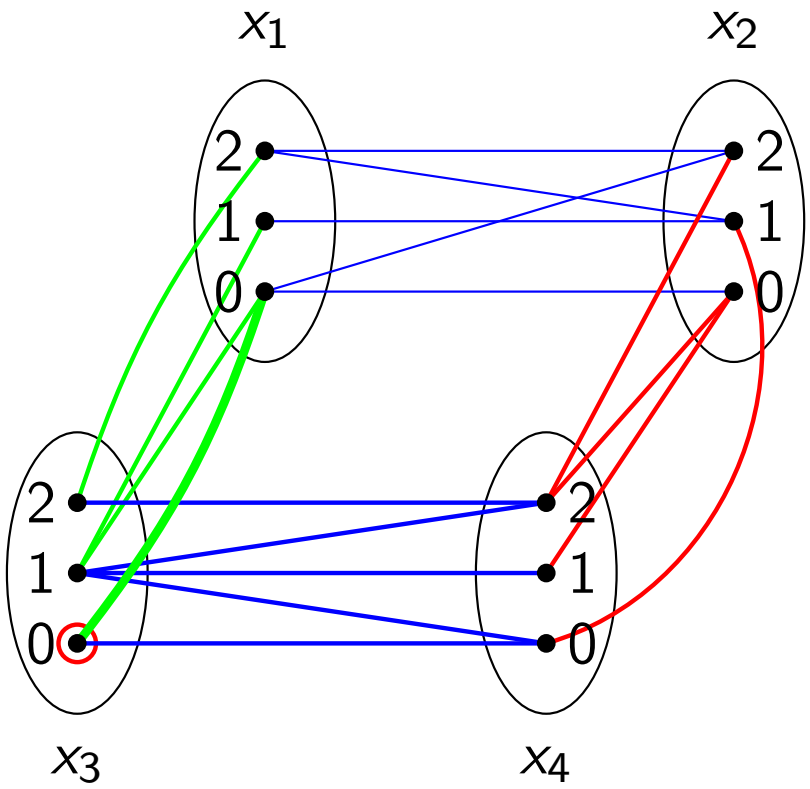
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



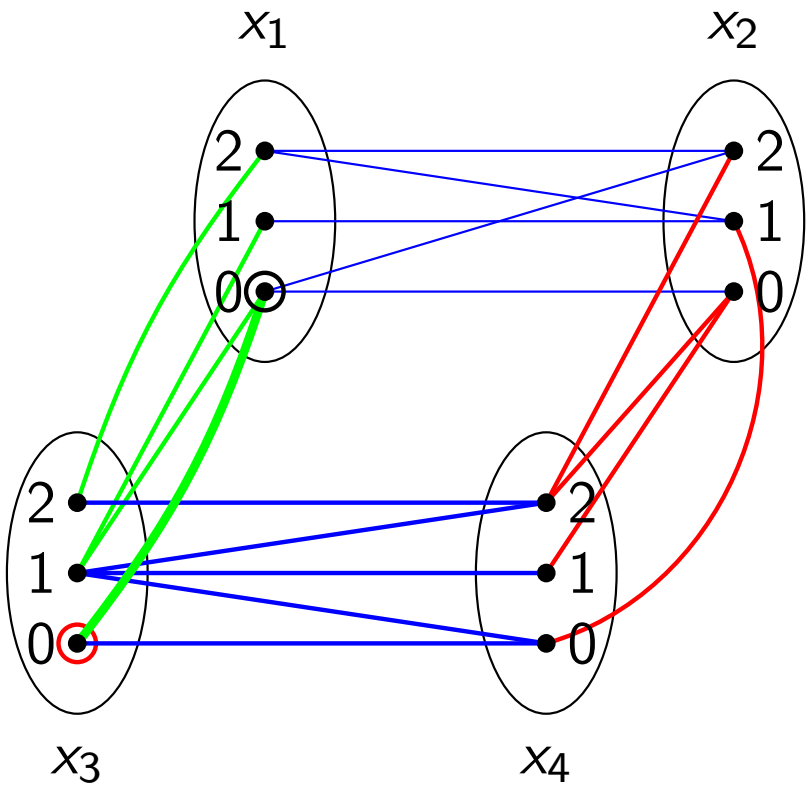
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



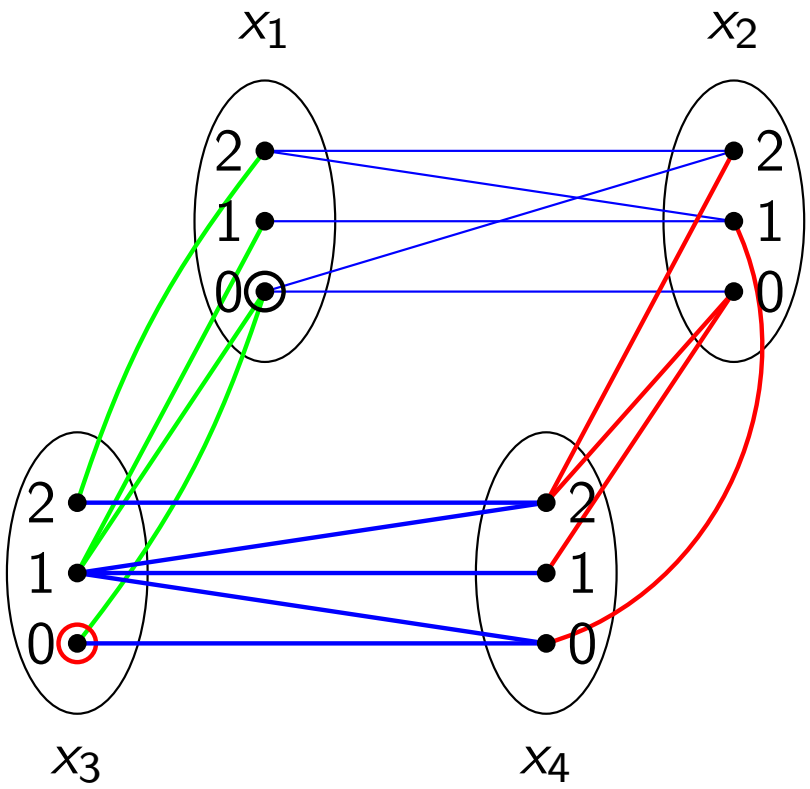
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



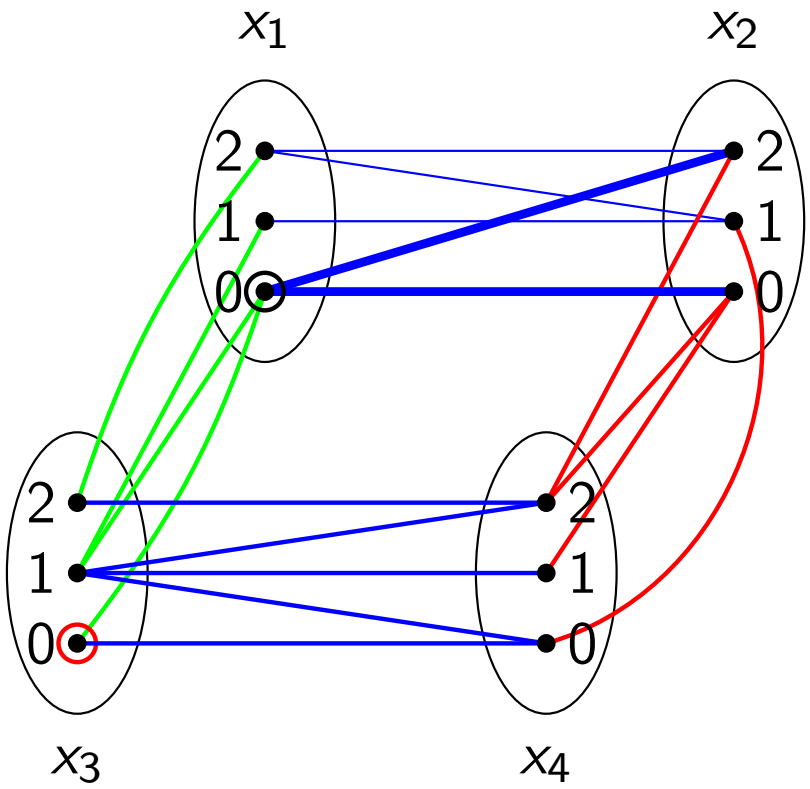
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



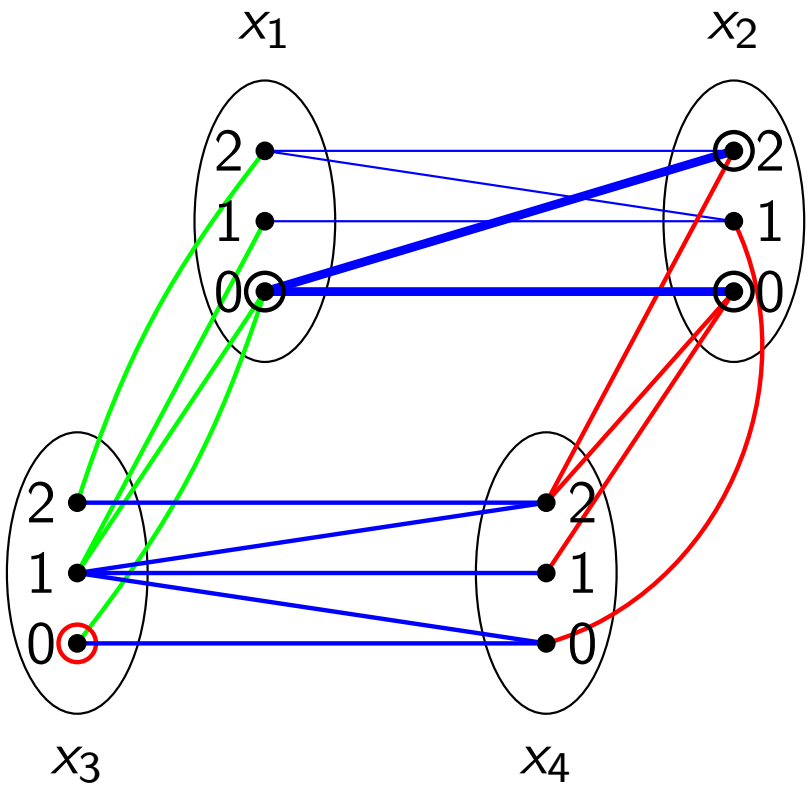
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



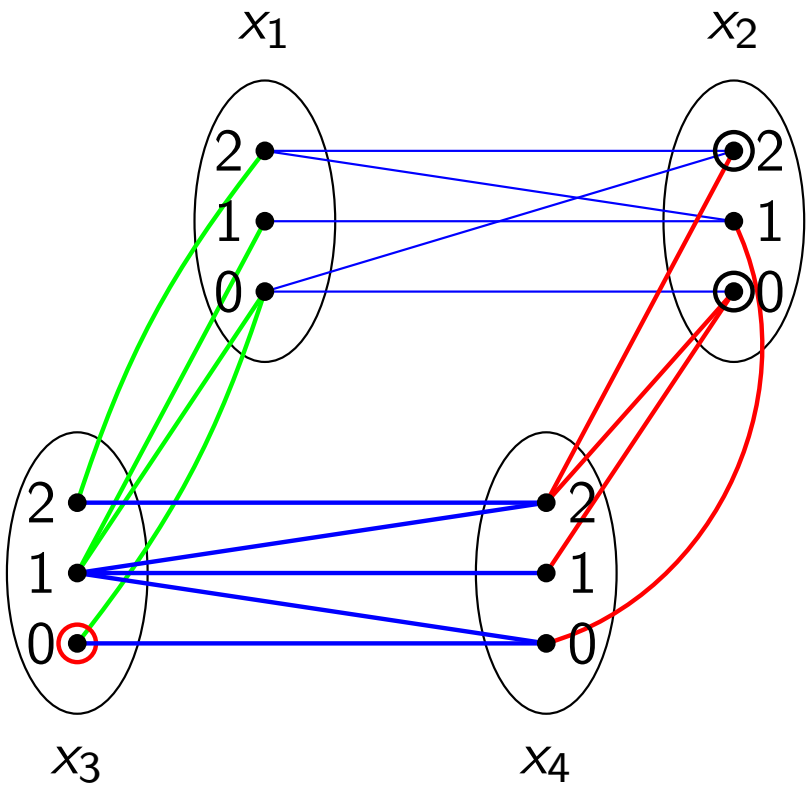
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



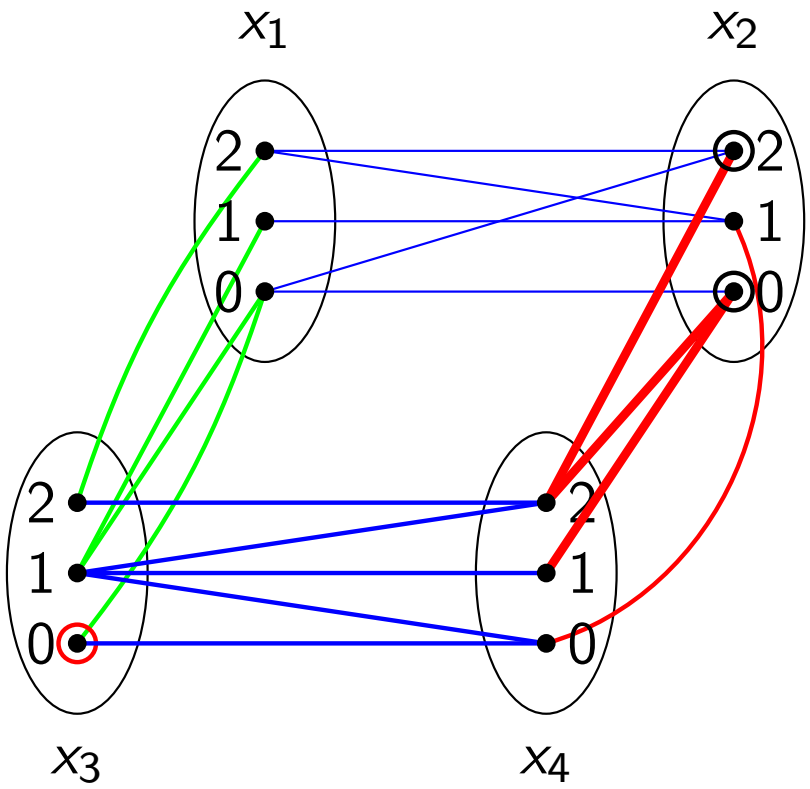
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



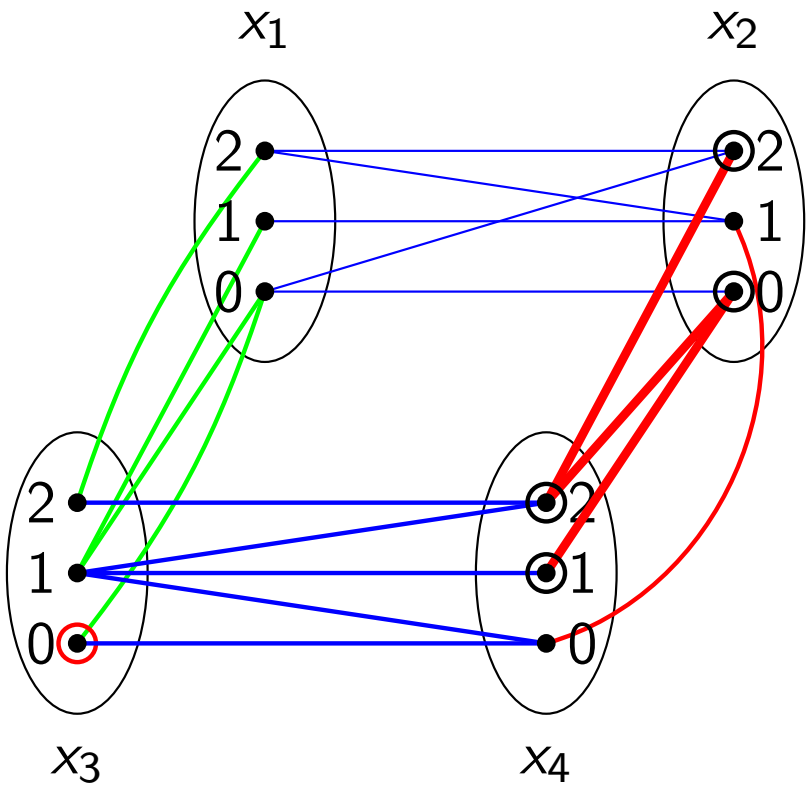
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



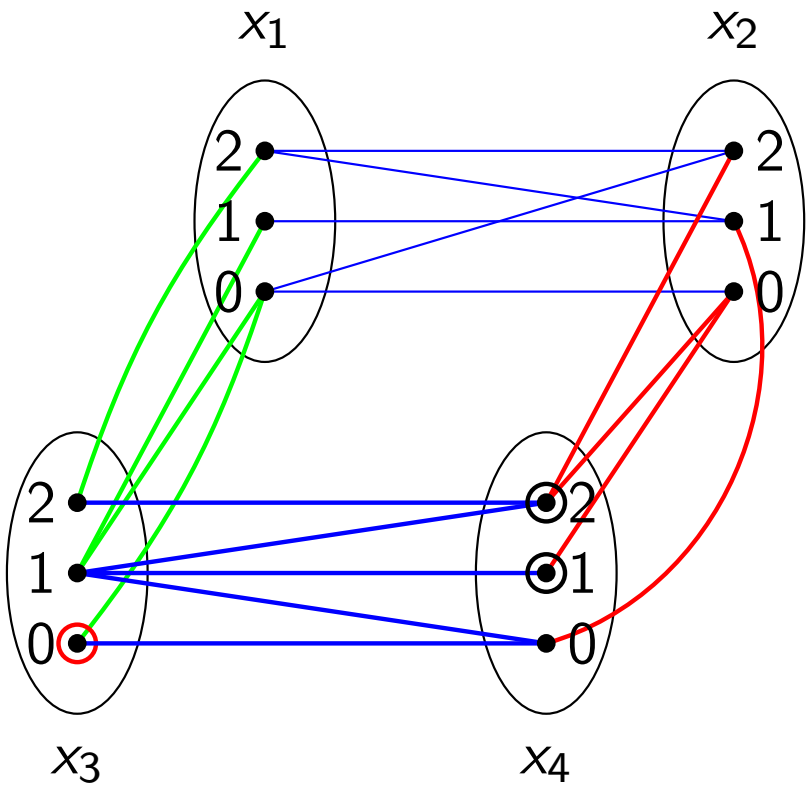
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



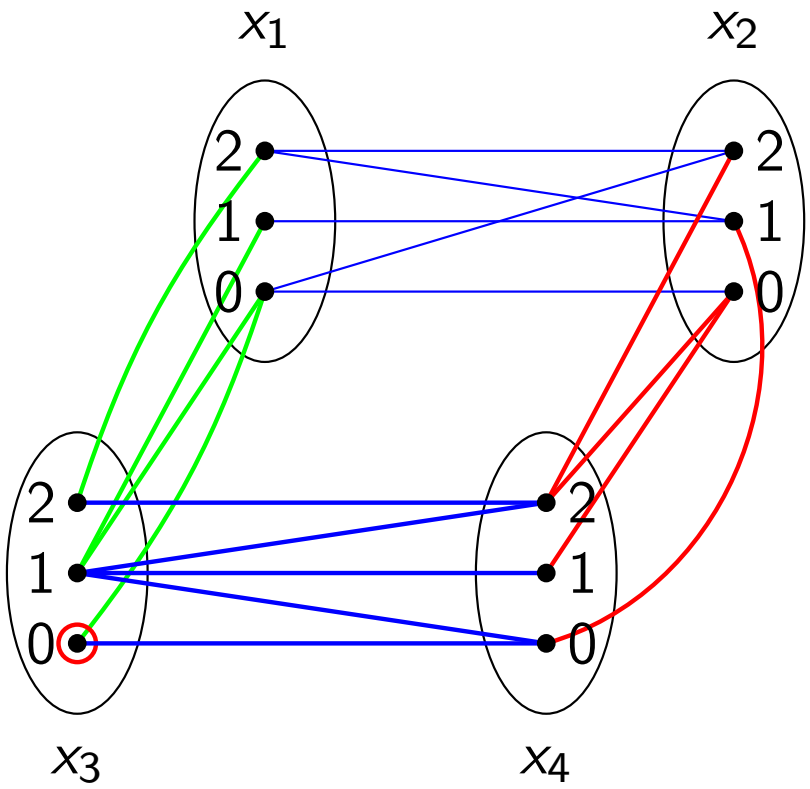
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



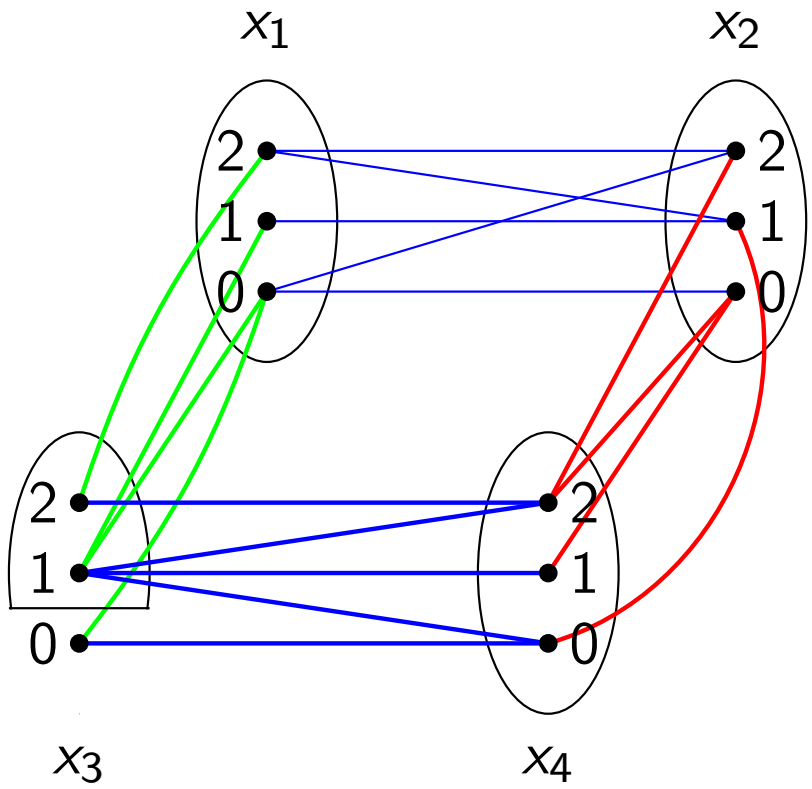
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



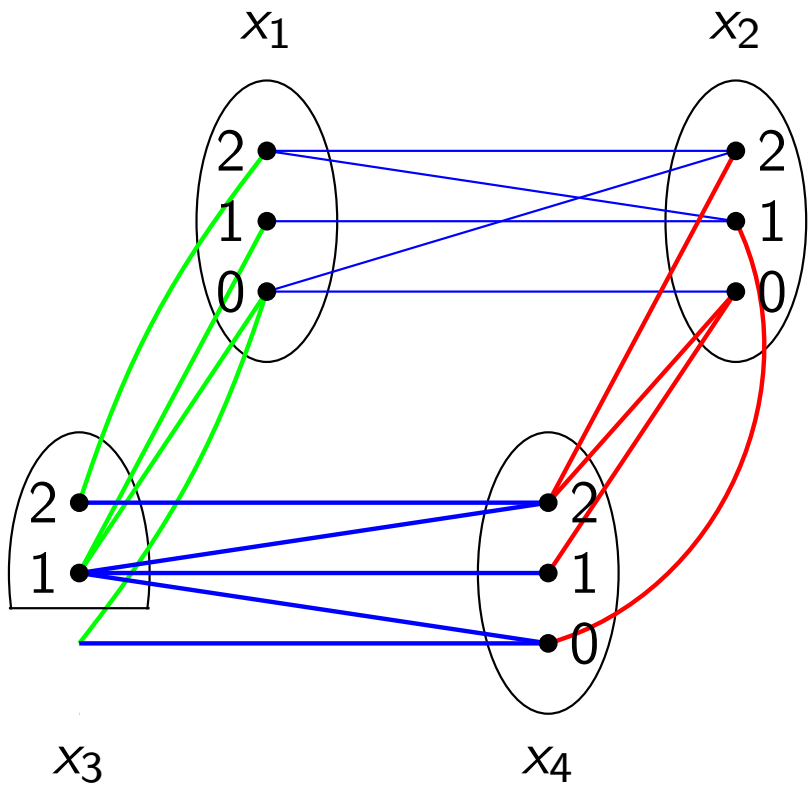
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



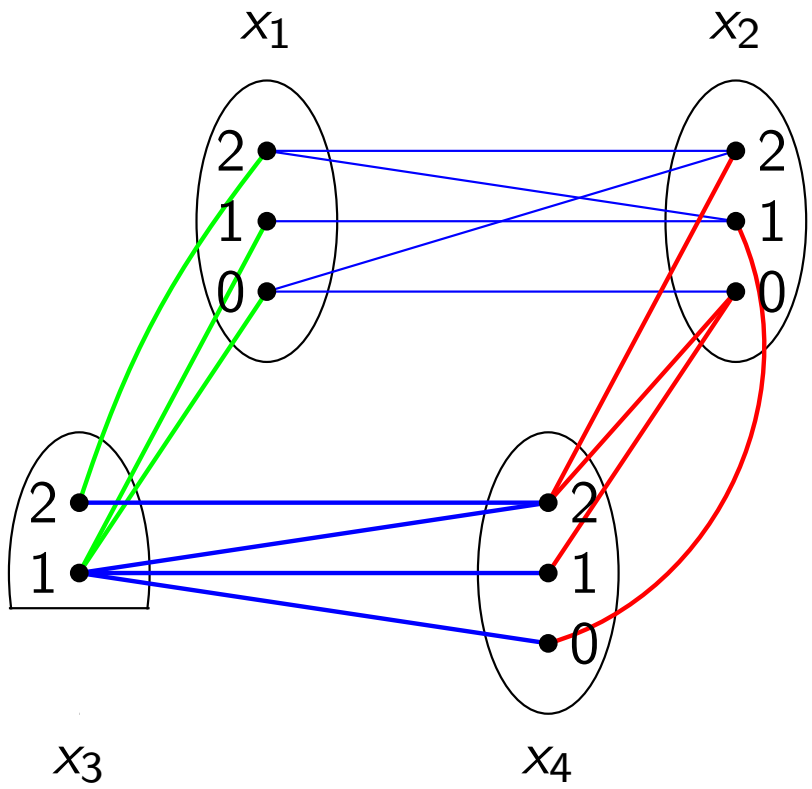
Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

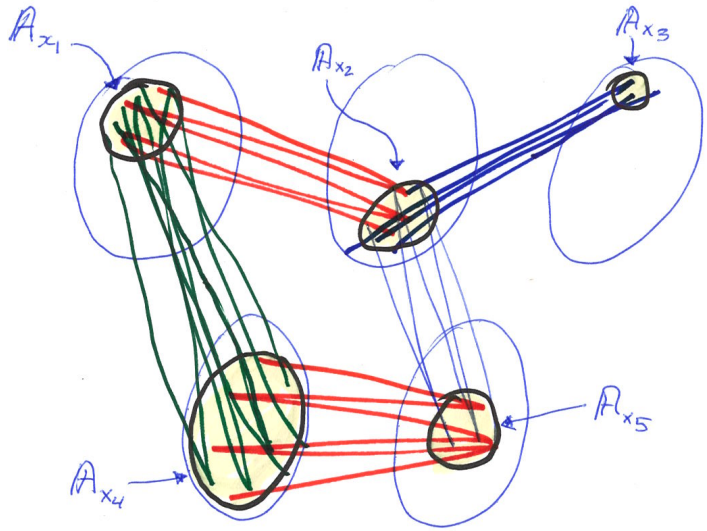
Given a CSP instance Θ and its potato diagram, there is a straightforward way to detect certain local inconsistencies, proving that some element of a potato can never belong to a solution.



Dynamically toss out elements discovered to not be in any solution; thus potatoes shrink; constraint relations are restricted to the new potatoes.

Two possible outcomes:

1. Some potato becomes empty. This proves the original CSP instance is inconsistent.
2. Potatoes stabilize at nonempty sets.



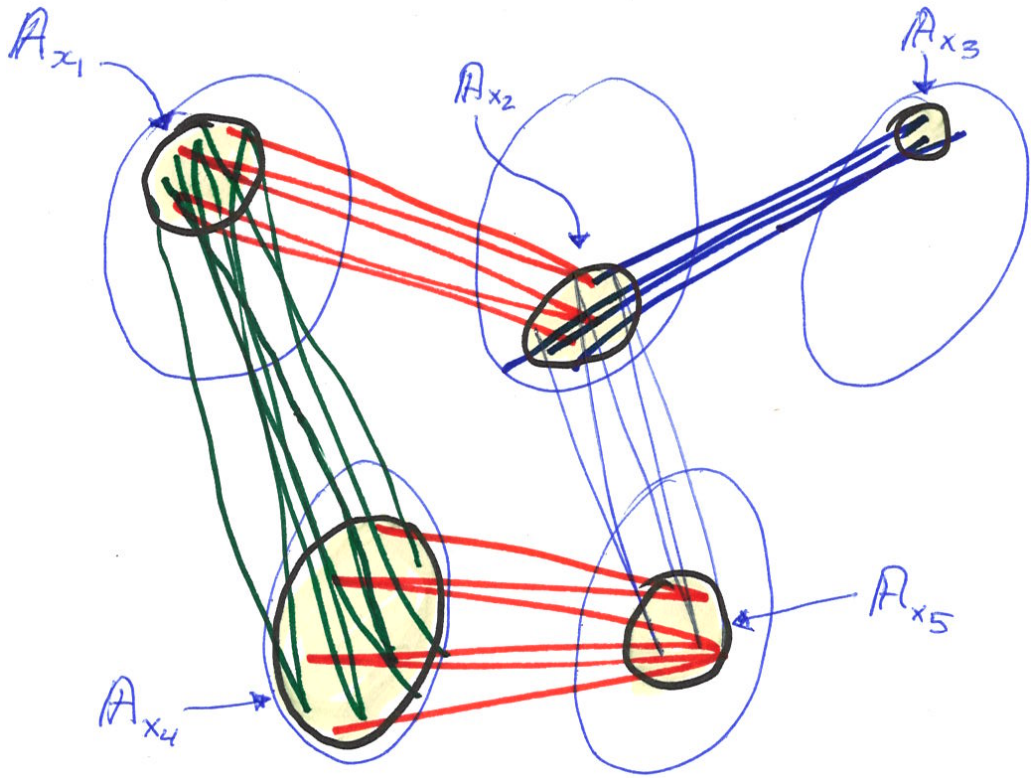
In this case:

- Let A_{x_i} denote the final potato at x_i. Then $A_{x_i} \leq \mathbb{A}$.
- For each original constraint $C_t = R_t(x_{i_1}, \ldots, x_{i_k})$, let the final constraint be $C'_t = R'_t(x_{i_1}, \ldots, x_{i_k})$. Then $R'_t \leq \mathbb{A}_{x_{i_1}} \times \cdots \times \mathbb{A}_{x_{i_k}}$.

The resulting

- family $(\mathbb{A}_{x_i} : 1 \leq i \leq n)$ of subalgebras of $\mathbb{A}$, and
- set $\{C'_t : 1 \leq t \leq m\}$ of compatible constraints on them,

is called a **multi-sorted** CSP instance **compatible with** $\mathbb{A}$.



It is no longer a CSP instance over the structure **A**, but we don't care.

The resulting multi-sorted CSP instance can be assumed to satisfy further "local consistency" properties, such as:

(1-consistency): Each constraint relation R'_t is subdirect in its final potatoes:

$$R'_t \leq_{sd} \mathbb{A}_{x_{i_1}} \times \cdots \times \mathbb{A}_{x_{i_k}}.$$

which means $\mathrm{proj}_{x_i}(R'_t) = A_{x_i}$ for all $i = 1, \ldots, k$.

(cycle-consistency): roughly, 1-consistency plus the following:

Any closed path from A_x to A_x through a sequence of constraints must be able to connect any point $a \in A_x$ to itself.

This is our problem:

- Given a finite (idempotent) Taylor algebra $\mathbb{A}$, and...
- A multi-sorted CSP instance Θ compatible with $\mathbb{A}$:
  - Potatoes $\mathbb{A}_x$ are subalgebras of $\mathbb{A}$, and
  - Constraints $R(x_1, \ldots, x_n)$ are subdirect subalgebras of their potatoes:

$$\mathbb{R} \leq_{sd} \mathbb{A}_{x_1} \times \cdots \times \mathbb{A}_{x_n}$$

- And assuming some higher level of local consistency (such as cycle-consistency):

To understand and organize the "reasons" that can make Θ inconsistent.

There is algebra here! (Come back Wednesday)