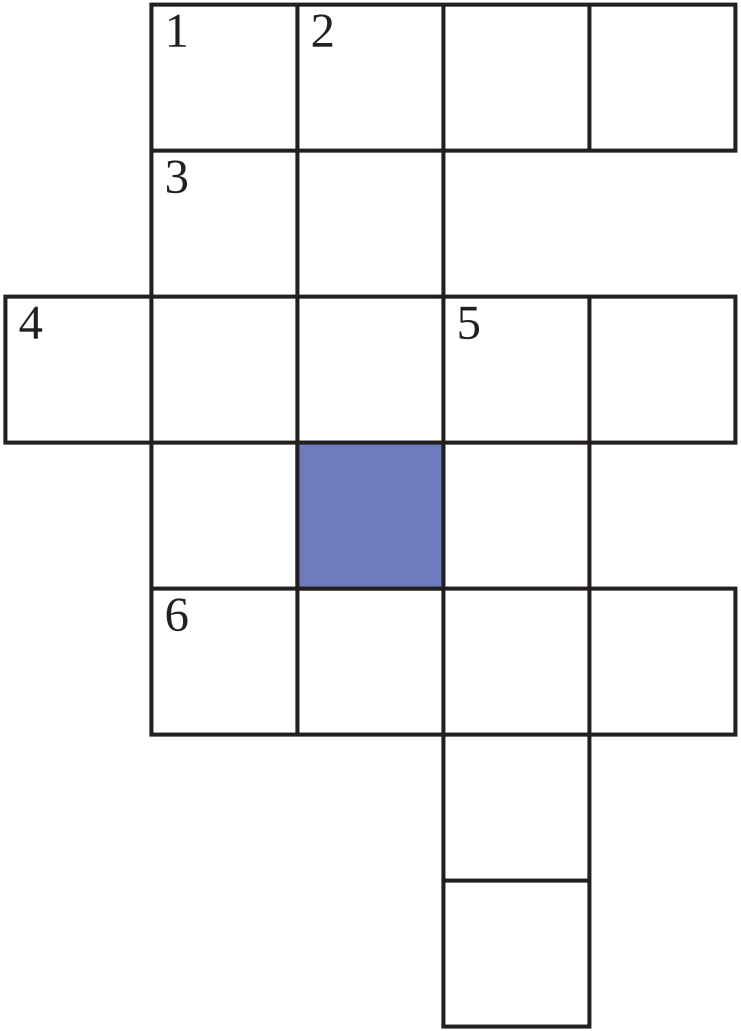
Constraint satisfaction revisited

- A Constraint Satisfaction problem consists of:
  - a set of variables
  - a set of possible values, a domain for each variable
  - A set of constraints amongst subsets of the variables (relations)
- The aim is to find a set of assignments that satisfies all constraints, or to find all such assignments.

Example: crossword puzzle



at, be, he, it, on,
eta, hat, her, him,
one,
desk, dove, easy,
else, help, kind,
soon, this,
dance, first, fuels,
given, haste,
loses, sense,
sound, think,
usage

Dual Representations

Two ways to represent the crossword as a CSP

- First representation:
  - nodes represent the positions 1 to 6
  - domains are the words
  - constraints specify that the letters on the intersections must be the same.
- Dual representation:
  - nodes represent the intersecting squares
  - domains are the letters
  - constraints specify that the words must fit

Representations for image interpretation

- First representation:
  - nodes represent the chains and regions
  - domains are the scene objects
  - constraints correspond to the intersections and adjacency
- Dual representation:
  - nodes represent the intersections
  - domains are the intersection labels
  - constraints specify that the chains must have same marking

Arc Consistency for non-binary relations

- Each relation $R(X_1, \ldots, X_k)$ converted into k hyperarcs:

$$\langle X_1, R(X_1, \ldots, X_k) \rangle$$

$$\ldots$$

$$\langle X_k, R(X_1, \ldots, X_k) \rangle$$

- Hyperarc $\langle X_i, R(X_1, \ldots, X_k) \rangle$ is **arc consistent** if
  - for every $v_i \in domain(X_i)$
  - there exists $v_1 \in domain(X_1)$, ... $v_{i-1} \in domain(X_{i+1})$, $v_{i+1} \in domain(X_{i+1})$ ... $v_k \in domain(X_k)$
  - such that $R(X_1, \ldots, X_k)$ is true.

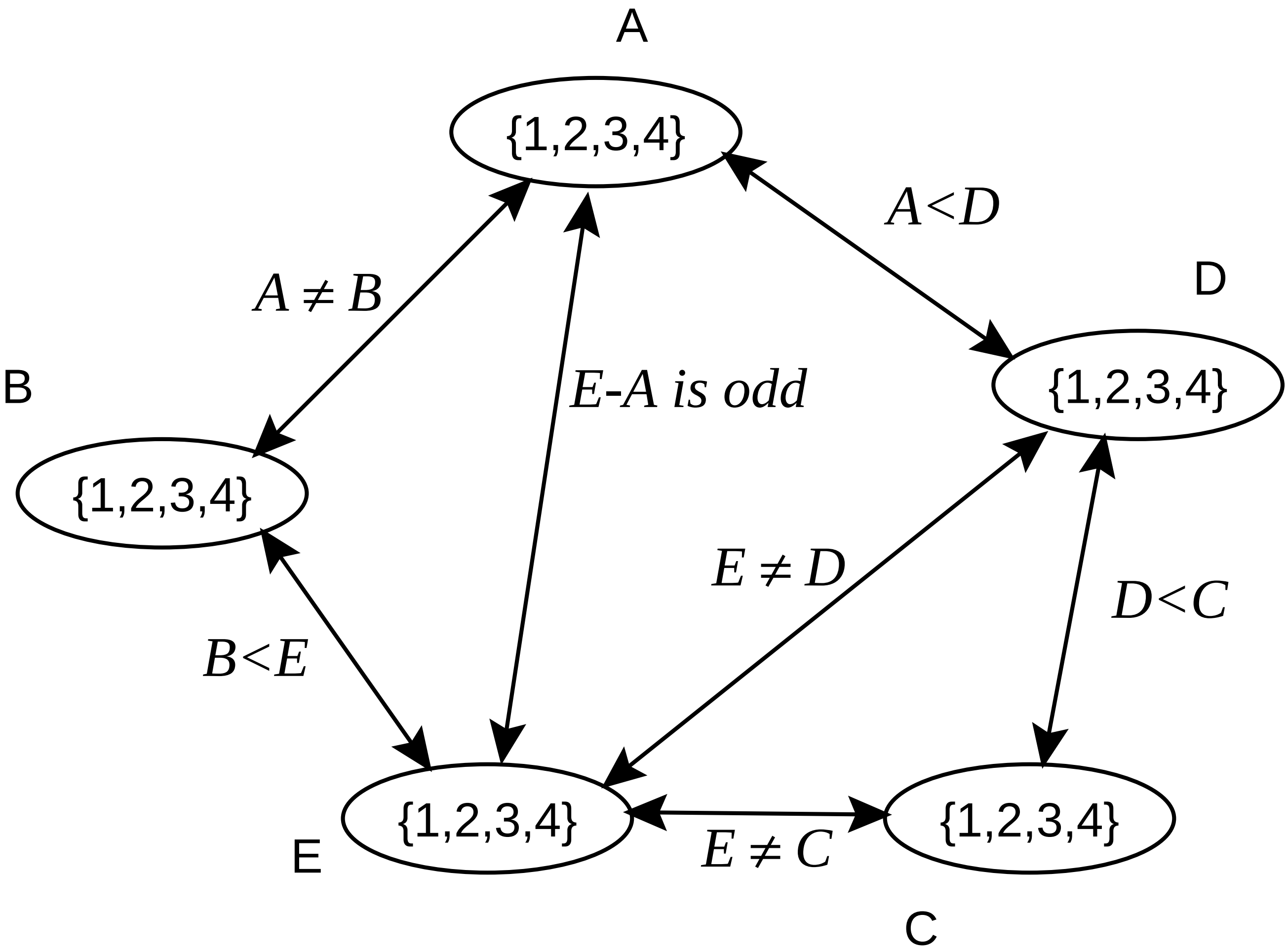
Variable Elimination

- Idea: eliminate the variables one-by-one passing their constraints to their neighbours
- To eliminate a variable X_i:
  - Join all of the relations in which X_i appears.
  - Project the join onto the other variables, forming a new relation.
  - Remember which values of X_i are associated with the tuples of the new relation.
  - Replace the old relations containing X_i with the new relation.

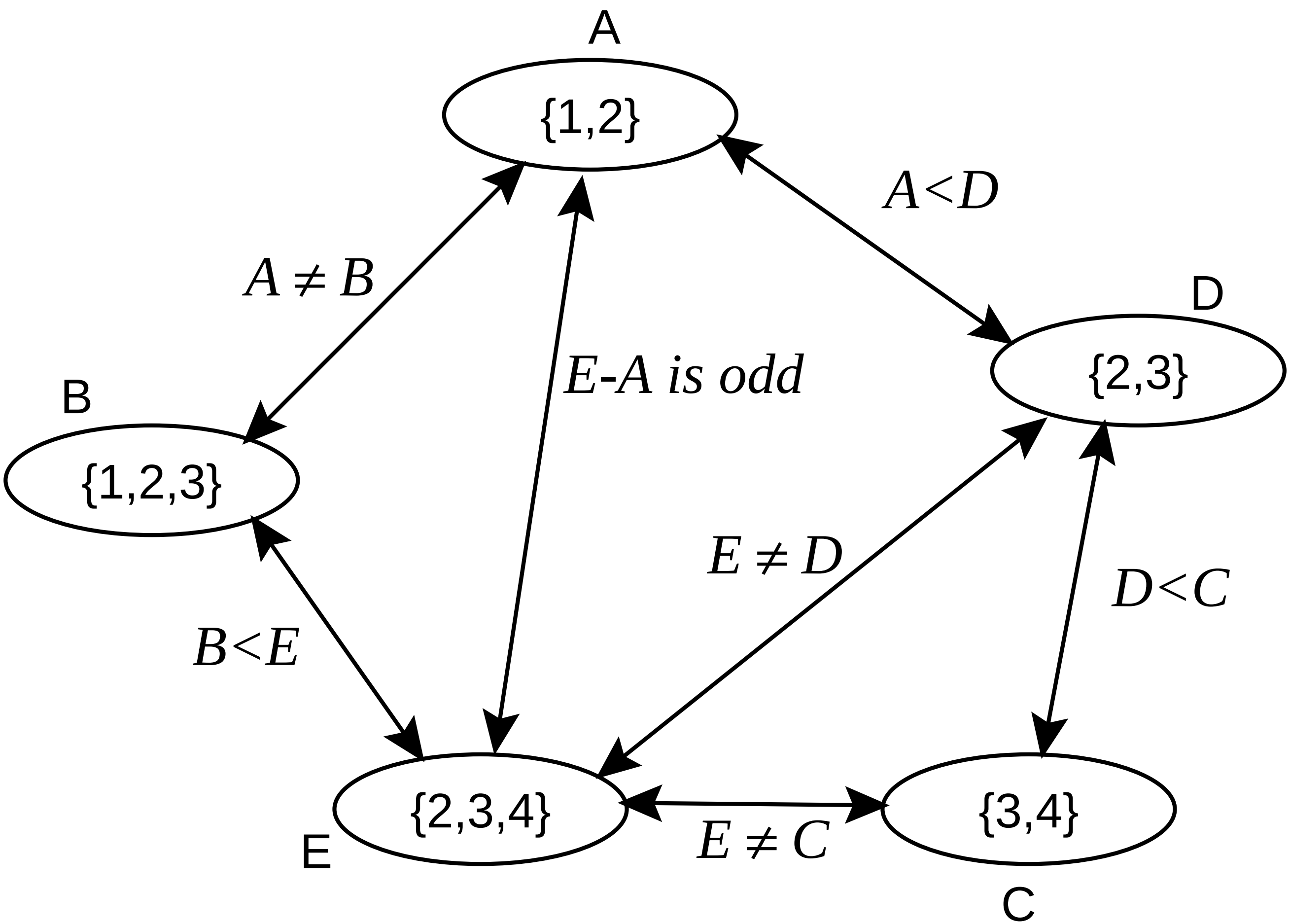
Variable elimination (cont.)

- When there is a single variable remaining, if it no values, the network was inconsistent.
- The solutions can be computed from the remembered mappings.
- The variables are eliminated according to some elimination ordering
- Different elimination odering result in different size relations being generated.

Example network



Example: arc-consistent network



Example: eliminating C

$r_1 : C \neq E$	C	E
	3	2
	3	4
	4	2
	4	3

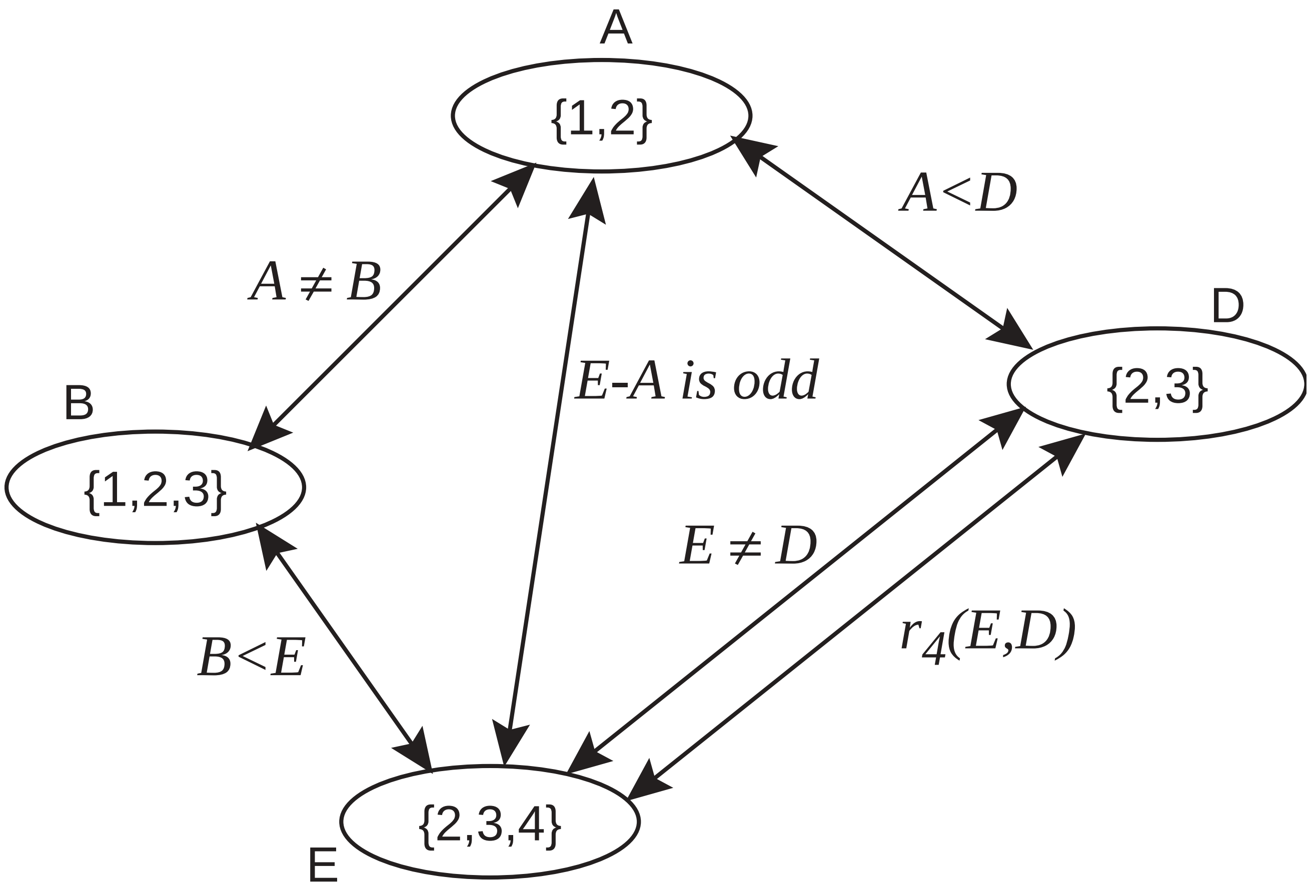
$r_2 : C > D$	C	D
	3	2
	4	2
	4	3

$r_3 : r_1 \bowtie r2$	C	D	E
	3	2	2
	3	2	4
	4	2	2
	4	2	3
	4	3	2
	4	3	3

$r_4 : \pi_{\{D,E\}} r_3$	D	E
	2	2
	2	3
	2	4
	3	2
	3	3

$\hookrightarrow$ new constraint

Resulting network after eliminating *C*



Stochastic local search for CSPs

- The following can be used to solve CSPs:
  - hill climbing on the assignments.
    - Choose the best variable then the best value.
    - Choose the best variable-value pair

    Best: satisfies the most constraints
  - random assignments of values.
  - random walks
- A mix works even better.

Evaluating Algorithms

- Summary statistics such as mean or median of run times are often not useful in comparing algorithms.
- The information about an algorithm performance can be determined from a runtime distribution.
- A runtime distribution specifies the proportion of the instances that have a running time less than any particular run time.