Constraint Logic: [Hearn & Demaine 2009]

**Constraint graph** (a model of computation)

- **graph with red & blue edges** $\rightarrow$ **MACHINE**
  - weight 1
  - weight 2
- **orientation of edges** $\rightarrow$ **CONFIGURATION**
  - such that incoming weight $\geq 2$ at each vertex
  - $\rightarrow$ **INFLOW CONSTRAINT**
- move = reversal of one edge
  - resulting in valid configuration (i.e. satisfying inflow constraint)
  - $\Rightarrow$ every move can be undone (immediately)
- asynchronous move: directed edge $\leftrightarrow$ undirected
- equivalent power [Viglietta - CCCG 2013]

Nondeterministic Constraint Logic: (NCL)

- given constraint graph
- find a sequence of moves
- goal 1: reverse specified edge
- goal 2: reach specified configuration

- PSPACE-complete, even for just 2 vertex types: AND $\triangleright$ & OR & even for planar graphs
AND vertex = 2 incident red edges → inputs + 1 incident blue edge → output
- output can activate = be directed out only if both inputs active = directed in edges are consistently (in)active from both ends:

- but there can be a delay between input activations & output activation

SPLIT vertex = 1 incident blue edge → input + 2 incident red edges → outputs
- outputs can activate only if input active
- alternative view of AND vertex

OR vertex = 3 incident blue edges → 2 inputs + 1 output
- output can activate only if at least one of the inputs active

NOT vertex is impossible:
- goal: output can activate only if input is not activated
  or: output never activated when input is
- inflow constraint always happier to have activated inputs & de-activated outputs
\text{CHOICE\ vertex} = 3 \text{ incident red edges} \\
\rightarrow 1 \text{ input} + 2 \text{ outputs} \\
- \text{output can activate only if} \\
\text{input active} & \text{other output not active} \\
- \text{gadget reduction to AND/OR} \\

\text{Red-blue conversion:} \\
- \text{needed for e.g. output of AND or OR (blue)} \\
\rightarrow \text{input of AND or CHOICE (red)} \\
- \text{gadget on pairs} \\
- \text{will force even \#} \\

\text{CNF formulas (ANDs of ORs)} \\
\text{via dual-rail logic for } x_i & \overline{x_i} \\
- \text{force at most one true via SPLIT} \\
- \text{output can activate only if} \\
\text{formula is satisfiable} \\

\text{Wire terminators:} \rightarrow \text{degree-1 vertices} \\
- \text{unconstrained blue & red terminators} \\
\text{(why red, instead of red-blue conversion?}\) \\
\text{to force equal \# red-blue conversions)} \\
- \text{forced-inward blue terminator} \\

\text{Constraint Graph Satisfaction:} \exists \text{ configuration?} \\
- \text{NP-complete} \\
\text{[Hearn]}
NCL is PSPACE-complete:
- reduction from CNF QSAT
- latch gadget - one bit of memory
  - "unlock" input & two outputs A & B
  - when locked, state is fixed:
    can output A or can output B (never both)
  - when unlocked, state is free to flip
    (and can output both A & B)
- existential quantifier gadget
  - latch to make guess
  - lock before activating rest of formula
- universal quantifier gadget
  - upper latch to set & lock variable
  - lower latch set up initially, (try-in inactive)
  - settleable down if x=0 & satisfied-in (⇔try-out)
  - satisfied-out only if latch down & x=1 & sat-in
- final satisfied-out flippable ⇔ formula true
- attach latch, flip, unwind ⇒ config-to-config.

Planar NCL is PSPACE-complete
- crossover gadget:
  - B can point down ⇔ A can ⇔ I can
  - D can point right ⇔ C can ⇔ E can
  - to cross red edges: convert to blue & back
  - vertex with 4 red edges:
    ≥ 2 edges must face inward
Grid constraint graphs:
- $2 \times 2$ & $2 \times 3$ filler gadgets (all active)
- straight, turn, AND/OR gadgets

Protected OR: guaranteed $\leq 1$ input activated
- can build OR
- use of red-blue conversion OR (forced config)

Reconfiguration 3SAT:
- given 2 satisfying assignments to 3CNF formula
- move = flip one variable false $\Rightarrow$ true
- does move sequence from one assignment to other? [Gopalan, Kalai-itis, Maneva, Papadimitriou - SICOMP 2009]
- PSPACE-complete
- easy reduction from NCL: [Eisenstat 2014]
  - edge $\rightarrow$ variable (0/1 indicates orientation)
  - OR vertex $\xrightarrow{y}$ $(x \text{ in}) \lor (y \text{ in}) \lor (z \text{ in})$
  - AND vertex $\xrightarrow{y}$ $(x \text{ out} \Rightarrow y \text{ in}) \land (x \text{ out} \Rightarrow z \text{ in})$ (2CNF)
  - formula = AND of all these clauses
- NCL is essentially a special case of this problem
- other reconfiguration problems:
  [Ito, Demaine, Harvey, Papadimitriou, Sideri, Uehara, Uno - TCS 2011]
Sliding-block puzzles: (initial motivation)
- rectangular blocks in rectangular box
- move = noncolliding slide
- goal: move one block, e.g. out hole of box
- PSPACE-complete even for 1x2 blocks
  \[\text{[Hearn & Demaine 2002]}\]

Sliding tokens = reconfig. Independent Set
- like 1x1 blocks on a graph but require no adjacent tokens

Rush Hour: \[\text{[Flake & Baum 2002; Hearn & Demaine 2002]}\]
- blocks can only slide in long direction
- PSPACE-complete for 1x2 & 1x3
- 1x2 PSPACE-complete \[\text{[Tromp & Cilibrasi 2008]}\]
- 1x1 OPEN
- triangular PSPACE-complete

Hinged dissection: chain of blocks folding
- polygon A \(\rightarrow\) polygon B
- always exist, avoiding collisions
  \[\text{[Abbott, Abel, Charlton, Demaine, Demaine, Kominers - DCG 2008]}\]
- polyabolo font - collisions?
- avoiding collisions is PSPACE-complete
  \[\text{[Hearn & Demaine]}\]
**Sokoban:** [Culberson 1998; Hearn & Demaine 2002]
- PSPACE-complete
- most blocks where they need to be
- goal: satisfy formula, move 1 block, unwind
- can't wedge a block immovable
- AND/OR gadgets
- parity fix via stretching
- tunnels to reach all areas
- turn gadget

**Push-2F:** [Demaine, Hearn, Hoffmann - CCCG 2002]
- lock gadget } enough for Viglietta framework
- crossover
- NCL AND/OR out of that

**Rolling block mazes:** [Holzer & Jacobi - FUN 2012]
- 1x1x2 blocks, which can “roll” onto clear space
- rectangular frame

**Plank puzzles / River Crossing** [Hearn 2004]
- player can traverse, pick up, drop planks
- can hold only one at a time
- planks must end on posts
- global traversal of gadgets via 3 length-3 planks
Dynamic map labeling: [Buchin & Gerrits – ISAAC 2013]
- want to reconfigure labels (squares) next to points while adding/panning/zooming map

Partial searchlight scheduling: [Viglietta – CCCG 2003]
- searchlight = rotatable ray around a point
- intruder can move super fast but not through a light ray
- want to guarantee a region within a polygon is intruder-free