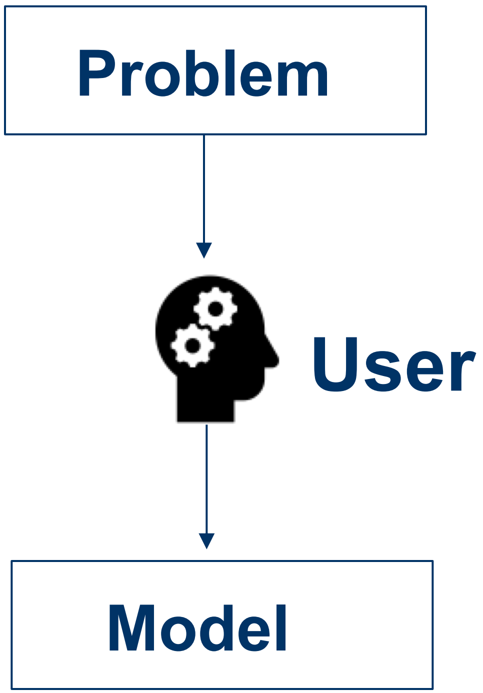
PART I: Modeling

Modeling in CP

- User models a decision problem by formalizing:
  - the unknowns of the decision → decision variables (X_i).
  - possible values for unknowns → domains ($D(X_i) = \{v_j\}$).
  - relations between the unknowns → constraints ($r(X_i, X_{i'})$).



Formalization as a Constraint Satisfaction Problem (CSP)

- A CSP is a triple **<X,D,C>** where:
  - **X** is a set of decision variables $\{X_1,...,X_n\}$;
  - **D** is a set of domains $\{D_1,...,D_n\}$ for **X**:
    - D_i is a set of possible values for X_i;
    - usually non-binary and assume finite domain;
  - **C** is a set of constraints $\{C_1,\ldots,C_m\}$:
    - C_i is a relation over $X_j,...,X_k$, denoted as $C_i(X_j, \ldots, X_k)$;
    - C_i the set of combination of allowed values $C_i \subseteq D(X_j) \times ... \times D(X_k)$.

- A solution to a CSP is an assignment of values to the variables which satisfies (that is feasible for) all constraints simultaneously.

Constraint Optimization Problems

- CSP enhanced with an optimization criterion, e.g.:
  - minimum cost;
  - shortest distance;
  - fastest route;
  - maximum profit.
- Formally, **<X,D,C,f>** where **f** is the formalization of the optimization criterion as an objective variable. Goal: minimize **f** (maximize **–f**).

Simple Examples

- Variables

  $X = \{X_1, X_2\}$
- Domains

  $D(X_1) = [1..3]$, $D(X_2) = [1..3]$
- Constraints
  - $C_1(X_1, X_2) = \{(1,2), (1,3), (2,3)\}$
  - $C_2(X_1, X_2) = \{(1,2), (2,3)\}$
- Solutions

  $X_1 = 1, X_2 = 2$

  $X_1 = 2, X_2 = 3$

- Variables

  $X = \{X_1, X_2, X_3\}$
- Domains

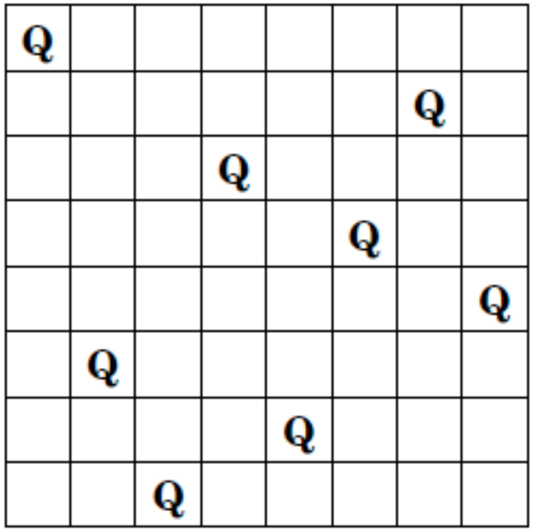
  $D(X_1) = D(X_2) = D(X_3) = \{1, 3, 5\}$
- Constraints
  - $X_1 + X_2 \leq X_3$
  - alldifferent($[X_1, X_2, X_3]$)
- Solutions

  $X_1 = 1, X_2 = 3, X_3 = 5$

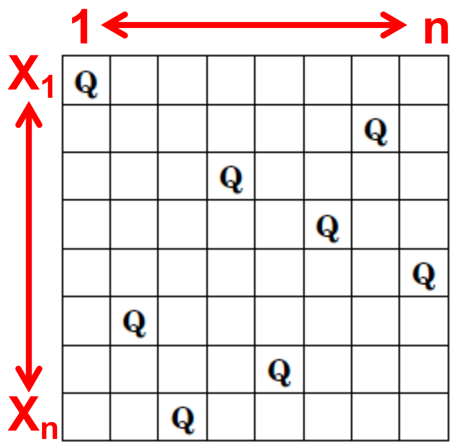
  $X_1 = 3, X_2 = 1, X_3 = 5$

N-Queens

- Place n queens in an nxn board so that no two queens can attack each other.



N-Queens

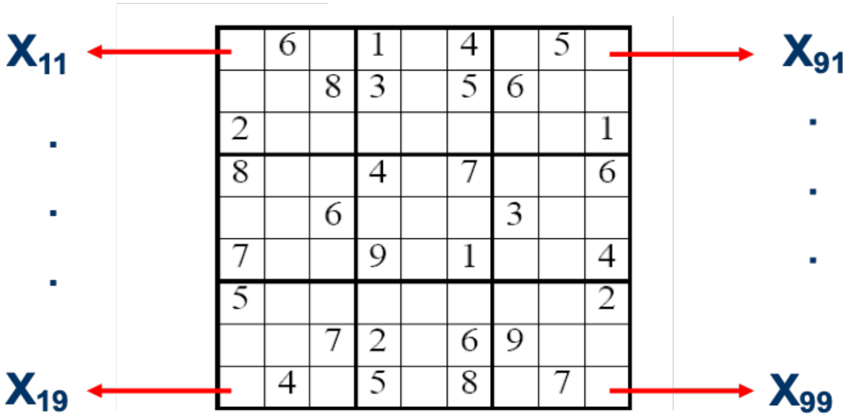


- Variables and Domains
  - A variable for each row $[X_1, X_2, \ldots, X_n]$ → no row attack
  - Domain values [1..n] represent the columns:
    - $X_i = j$ means that the queen in row i is in column j
- Constraints
  - alldifferent($[X_1, X_2, \ldots, X_n]$) → no column attack
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$ → no diagonal attack

Sudoku

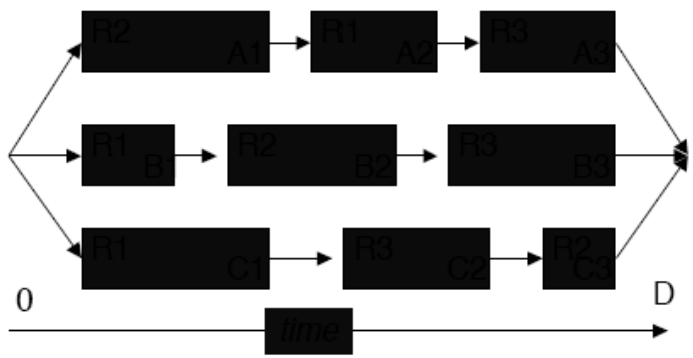
	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Sudoku



- Variables and Domains
  - 9x9 variables X_{ij} for each cell with domains [1..9].
    - X_{ij} = k means that the cell X_{ij} has the value k.
- Constraints
  - Initial assignments. E.g., $X_{21} = 6$.
  - Difference constraints on all the rows, columns, and 3x3 boxes. E.g.,
    alldifferent([X_{11}, X_{21}, X_{31}, …, X_{91}])
    alldifferent([X_{11}, X_{12}, X_{13}, …, X_{19}])
    alldifferent([X_{11}, X_{21}, X_{31}, X_{12}, X_{22}, X_{32}, X_{13}, X_{23}, X_{33}])

Task Scheduling



- Schedule n tasks on a machine, in time D, by obeying the temporal and precedence constraints:
  - each task t_i has a specific fixed processing time p_i;
  - each task t_i can be started after its release date r_i, and must be completed before its deadline d_i;
  - tasks cannot overlap in time;
  - precedence relations (→) must be respected.

Task Scheduling

- Variables and Domains
  - $Start_i$, representing the starting time of a task t_i, taking a value from [0..D].
  - Ensures that each task starts at exactly one time point.
- Constraints
  - Respect of release date and deadline
    - for all $i \in [1..n]$, $r_i \leq Start_i \leq d_i - p_i$
  - No overlap in time
    - for all $i < j \in \{1, \ldots, n\}$, $(Start_i + p_i \leq Start_j) \vee (Start_j + p_j \leq Start_i)$
  - Precedence constraints
    - $Start_i + p_i \leq Start_j$ for each pair of tasks $t_i \rightarrow t_j$

Optimal Map Colouring

- What is the minimum number of colours necessary to colour the neighbouring regions differently?

Optimal Map Colouring

- Variables and Domains
  - X_i for each of n regions with domain [1..n].
- Constraints
  - $X_i \neq X_j$ for each neighbour region i and j
- Objective variable
  - $f = \max(X_i)$
- Objective: minimize f

Variables and Domains

- Variable domains include the classical:
  - binary, integer, continuous.
- In addition, variables may take a value from *any* finite set.
  - e.g., X in {a,b,c,d,e}.
- There exist special "structured" variable types.
  - Set variables (take a set of elements as value).
  - Activities or interval variables (for scheduling applications).

Constraints

- Any kind of constraint can be expressed by listing all allowed combinations.
  - $C(X_1, X_2) = \{(0,0), (0,2), (1,3), (2,1)\}$
  - Extensional representation.
  - General but possibly inconvenient and inefficient with large domains.
- Declarative (invariant) relations among objects.
  - $X > Y$
  - Intensional representation.
  - More compact, clear but less general.

Properties of Constraints

- The order of imposition does not matter.
  - X + Y <= Z, Z >= X + Y
- Non-directional.
  - A constraint between X and Y can be used to infer domain information on Y given domain information on X and vice versa.
- Rarely independent.
  - Shared variables as communication mechanism between different constraints.

Constraints – Examples

- Algebraic expressions
  - $X_1 > X_2$
  - $X_1 + X_2 = X_3$
- Extensional constraints (table constraints)
  - (X,Y,Z) in {(a, a, a), (b, b, b), (c, c, c)}
- Variables as subscripts (element constraints)
  - Y = cost[X] (here Y and X are variables, ‘cost’ is an array of parameters)

Constraints – Examples

- Logical relations
  - $(X < Y) \vee (Y < Z) \rightarrow C$
- Global constraints
  - alldifferent($[X_1, X_2, X_3]$) instead of:

    $X_1 \neq X_2, X_1 \neq X_3, X_2 \neq X_3$
  - noOverlap($[Start_1, \ldots, Start_n], [p_1, \ldots, p_n]$) instead of:

    for all $i < j \in \{1, \ldots, n\}$, $(Start_i + p_i \leq Start_j) \vee (Start_j + p_j \leq Start_i)$
- Meta-constraints
  - $\sum_i (X_i > t_i) \leq 5$

Modeling is Critical!

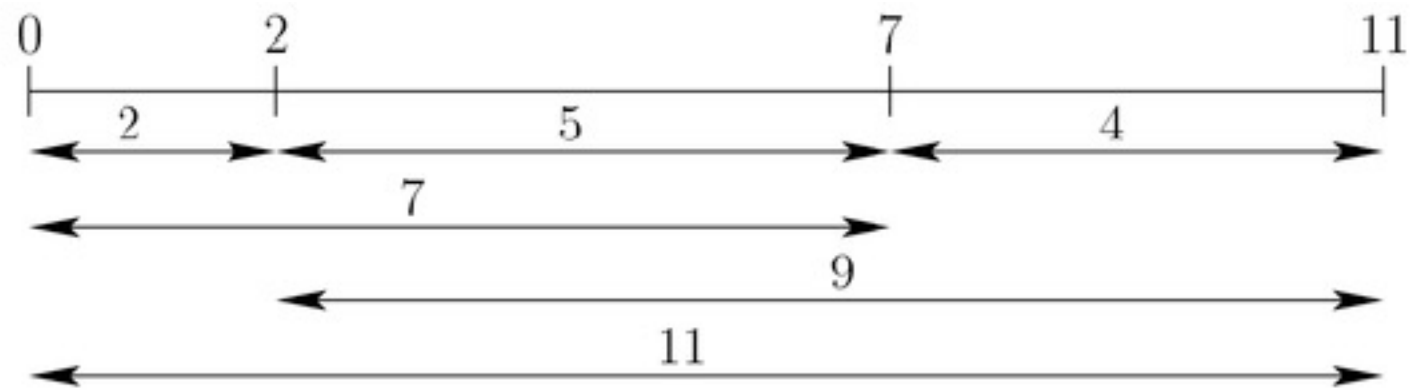
- Choice of variables and domains defines the search space size.
  - $|D(X_1)| \times |D(X_2)| \times \ldots \times |D(X_n)|$
  - Exponential in size!
- Choice of constraints defines:
  - how search space can be reduced;
  - how search can be guided.
- Need to go beyond the declarative specification!

Modeling is Critical

- Given the human understanding of a problem, we need to answer questions like:
  - which variables shall I choose?
  - which constraints shall I enforce?
  - can I exploit any global constraints?
  - do I need any auxiliary variables?
  - are some constraints redundant, therefore can be avoided?
  - are there any implied constraints?
  - can symmetry be eliminated?
  - are there any dual viewpoints?
  - among alternative models, which one shall I prefer?

Golomb Ruler

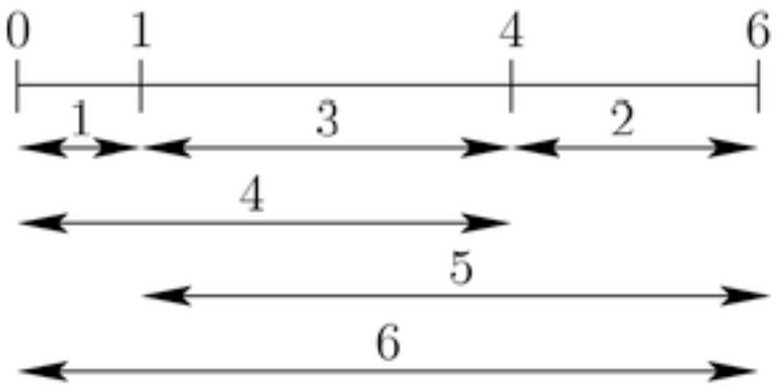
- Place m marks on a ruler such that:
  - distance between each pair of marks is different;
  - the length of the ruler is minimum.
- Applications in radio astronomy and information theory.
- Difficult to solve! Largest known ruler is of order 28.



A non optimal Golomb ruler of order 4.

Golomb Ruler

- Place m marks on a ruler such that:
  - distance between each pair of marks is different;
  - the length of the ruler is minimum.
- Applications in radio astronomy and information theory.
- Difficult to solve! Largest known ruler is of order 28.



An optimal Golomb ruler of order 4.

Naive Model

- Variables and Domains
  - $[X_1, X_2, .., X_m]$
  - X_i, representing the position of the i^{th} mark, taking a value from $\{0,1,\ldots,2^{(m-1)}\}$

Naive Model

- Variables and Domains
  - $[X_1, X_2, .., X_m]$
  - X_i, representing the position of the i^{th} mark, taking a value from $\{0,1,\ldots,2^{(m-1)}\}$
- Constraints
  - for all $i_1 < j_1$, $i_2 < j_2$, $i_1 \neq i_2$ or $j_1 \neq j_2$ $|X_{i1} - X_{j1}| \neq |X_{i2} - X_{j2}|$
- Objective: minimize $(\max([X_1, X_2, .., X_m]))$

Naive Model

- Variables and Domains
  - $[X_1, X_2, .., X_m]$
  - X_i, representing the position of the i^{th} mark, taking a value from $\{0,1,\ldots,2^{(m-1)}\}$
- Constraints
  - for all $i_1<j_1$, $i_2<j_2$, $i_1 \neq i_2$ or $j_1 \neq j_2$ $|X_{i1} - X_{j1}| \neq | X_{i2} - X_{j2}|$
- Objective: minimize $(\max([X_1, X_2, .., X_m]))$
- Problematic model.
  - $O(m^4)$ quaternary constraints.
  - Loose reduction in domains.

Better Model

- Auxiliary Variables
  - New variables introduced into a model, because either:
    - it is difficult/impossible to express some constraints on the main decision variables;
    - or some constraints on the main decision variables do not lead to significant domain reductions.
  - for all i<j D_{ij}, representing the distance between i^{th} and the j^{th} marks.
- Constraints
  - for all i<j, $D_{ij} = |X_i - X_j|$
  - alldifferent($[D_{12}, D_{13}, \ldots, D_{(m-1)m}]$)

Better Model

- Constraints
  - for all i<j $D_{ij} = |X_i - X_j|$
  - alldifferent($[D_{12}, D_{13}, \dots, D_{(m-1)m}]$)

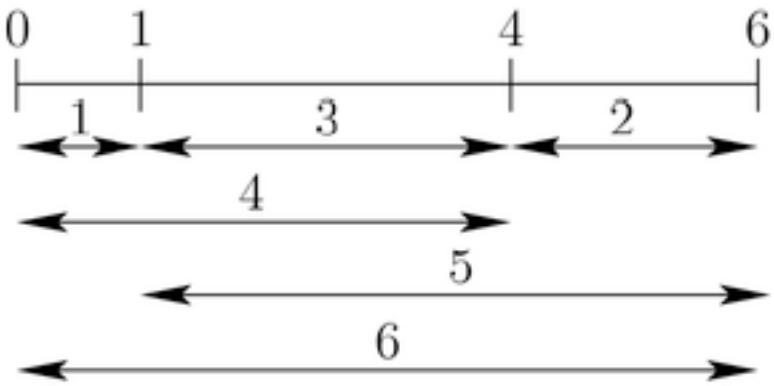
- Improvements
  - Quadratic $O(m^2)$ ternary constraints.
  - A global constraint.

Better Model

- Constraints
  - for all $i<j$ $D_{ij} = |X_i - X_j|$
  - alldifferent($[D_{12}, D_{13}, \ldots, D_{(m-1)m}]$)
  - alldifferent($[X_1, X_2, \ldots, X_m]$)
- Improvements
  - $O(m^2)$ ternary constraints.
  - A global constraint.
  - Implied constraint
    - Logically implied by the constraints of the problem which cannot be deduced by the solver.
    - Semantically redundant (no change in the set of solutions), computationally significant (can greatly reduce the search space)!

Improved Model

- Deducing information from Golomb Rulers of smaller order
  - If you consider any k consecutive marks of a Golomb Ruler of order n > k, they form a Golomb Ruler of order k.



An optimal Golomb ruler of order 4.

Improved Model

- Deducing information from Golomb Rulers of smaller order
  - If you consider any k consecutive marks of a Golomb Ruler of order $n > k$, they form a Golomb Ruler of order k.
  - Therefore, they must span over a distance at least as long as the optimal size of Rulers of order k.
  - for all $i<j$ $D_{ij} \geq$ optimal value of the ruler of order $(j-i+1)$

Symmetry in CSPs

- Creates many symmetrically equivalent search states:
  - A state leading to a solution/failure will have many symmetrically equivalent states.
- Bad when proving optimality, infeasibility or looking for all solutions.
  - May lead to thrashing.
- Variable and value symmetry.

Symmetries and Permutation

- Permutation
  - Defined over a discrete set S as a 1-1 function $\pi: S \rightarrow S$.
  - Intuitively: re-arrangement of a set of elements, e.g.,
    - i: 1 2 3 4 5
    - π(i): 3 5 4 2 1
- Variable Symmetry
  - A permutation π of the variable indices s.t. for each (un)feasible (partial) assignment, we can re-arrange the variables according to π and obtain another (un)feasible (partial) assignment.
  - Intuitively: permuting variable assignments.
  - π identifies a specific symmetry.

Variable Symmetries in Golomb Ruler

- Permuting variable assignments

  $X_1 = 0, X_2 = 1, X_3 = 4, X_4 = 6$

  $X_1 = 0, X_2 = 1, X_3 = 6, X_4 = 4$

  $X_1 = 0, X_2 = 4, X_3 = 1, X_4 = 6$

  $X_1 = 0, X_2 = 4, X_3 = 6, X_4 = 1$

  $X_1 = 0, X_2 = 6, X_3 = 1, X_4 = 4$

  $X_1 = 0, X_2 = 6, X_3 = 4, X_4 = 1$

  …
- m! permutations → m! variable symmetries
- For a given (un)feasible assignment, there are m! (un)feasible assignments.

Value Symmetry

- Value Symmetry
  - A permutation π of values s.t. for each (un)feasible (partial) assignment, we can re-arrange the values according to π and obtain another (un) feasible (partial) assignment.
  - Intuitively: permuting values.
  - π identifies a specific symmetry.

A Value Symmetry in Golomb Ruler

- Values can be permuted as:

  0 → 0, 1 → 2, 2 → 1, 3 → 3, 4 → 5, 5 →4 , 6 → 6

  (reversing the ruler)

  $X_1 = 0, X_2 = 1, X_3 = 4, X_4 = 6$ →

  $X_1 = 0, X_2 = 2, X_3 = 5, X_4 = 6$

Any other value symmetry in the models we have seen so far?

Variable and Value Symmetry

- Composition of a variable and a value symmetry.
- Golomb Ruler
  - Both variable assignments and values can be permuted.

    $X_1 = 0, X_2 = 1, X_3 = 4, X_4 = 6 \rightarrow X_1 = 0, X_2 = 2, X_3 = 5, X_4 = 6$
    $\rightarrow X_1 = 2, X_2 = 0, X_3 = 6, X_4 = 5$

  - For a given (un)feasible assignment, there are 2*m! (un)feasible assignments.

Symmetry Breaking Constraints

- Reduce the set of solutions and search space!
- Not logically implied by the constraints of the problem.
- Common technique: impose an ordering to avoid permutations.
  - E.g., $X_1 \leq X_2 \ldots. \leq X_n$ when $[X_{1,} X_{2, \ldots.,} X_n]$ are all symmetric.
- Attention: at least one solution from each set of symmetrically equivalent solutions must remain.

Improved Model

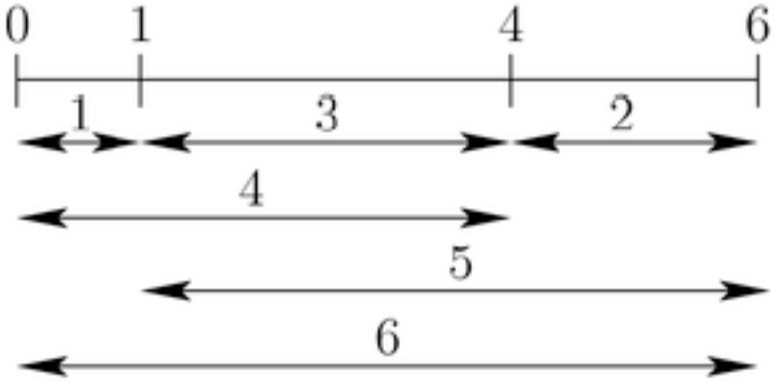
- Symmetry Breaking Constraints
  - $X_1 < X_2 < \ldots < X_m$
  - $X_1 = 0$
  - $D_{12} < D_{(m-1)m}$
- New Objective
  - minimize (X_m)

Improved Model

- Symmetry breaking constraints enable constraint simplification.
  - $X_1 < X_2 < \ldots < X_m$
    - alldifferent($[X_1, X_2, \ldots, X_m]$) becomes redundant (semantically and computationally).
    - for all $i<j$, $D_{ij} = |X_i - X_j|$ becomes for all $i<j$, $D_{ij} = X_j - X_i$
  - Note the terminology redundant vs implied.

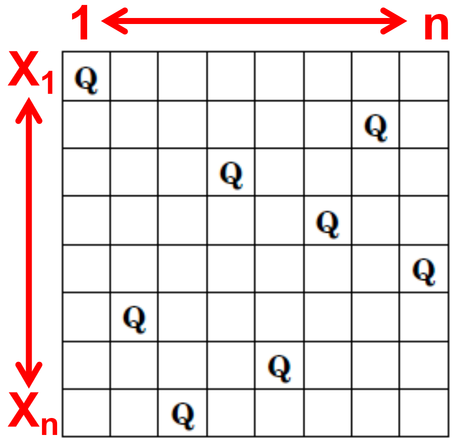
Improved Model

- Symmetry breaking constraints enable additional implied constraints.
  - for all i<j<k, $D_{ij} < D_{ik}$ and $D_{jk} < D_{ik}$
    $D_{ik} = D_{ij} + D_{jk}$



An optimal Golomb ruler of order 4.

Can We Improve This Model Too?



- Variables and Domains
  - A variable for each row $[X_1, X_2, \ldots, X_n]$ → no row attack
  - Domain values {1,...,n} represent the columns:
    - $X_i = j$ means that the queen in row i is in column j
- Constraints
  - alldifferent($[X_1, X_2, \ldots, X_n]$) → no column attack
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$ → no diagonal attack

N-Queens

- Diagonal attack constraint
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$

≡ for all $i<j$ $X_i - X_j \neq i - j$ and $X_i - X_j \neq j - i$ and
$X_j - X_i \neq i - j$ and $X_j - X_i \neq j - i$

≡ for all $i<j$ $X_i - i \neq X_j - j$ and $X_i + i \neq X_j + j$

≡ alldifferent($[X_1 - 1, X_2 - 2, \ldots, X_n - n]$)

≡ alldifferent($[X_1 + 1, X_2 + 2, \ldots, X_n + n]$)

A Better Model

- Original Model
  - alldifferent($[X_1, X_2, \ldots, X_n]$) → no column attack
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$ → no diagonal attack

- Alldiff Model
  - alldifferent($[X_1, X_2, \ldots, X_n]$)
  - alldifferent($[X_1 + 1, X_2 + 2, \ldots, X_n + n]$)
  - alldifferent($[X_1 - 1, X_2 - 2, \ldots, X_n - n]$)

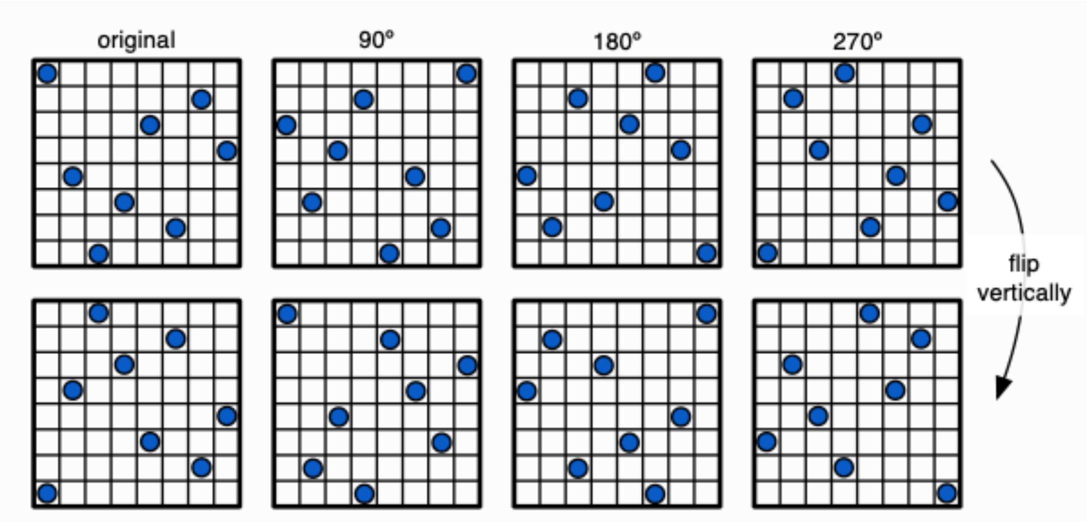
Modeling is Critical!

- Given the human understanding of a problem, we need to answer questions like:
  - which variables shall I choose?
  - which constraints shall I enforce?
  - can I exploit any global constraints?
  - do I need any auxiliary variables?
  - are some constraints redundant, therefore can be avoided?
  - are there any implied constraints?
  - can symmetry be eliminated?
  - are there any dual viewpoints?
  - among alternative models, which one shall I prefer?

Dual Viewpoint

- Viewing a problem P from different perspectives may result in different models for P.
- Each model yields the same set of solutions.
- Each model exhibits in general a different representation of P.
  - Different variables.
  - Different domains.
  - Different constraints.
    - Different size of the search space!
- Can be hard to decide which is better!

Symmetries of N-Queens



- Geometric symmetries.
  - Cannot impose an ordering like $X_1 \leq X_2 \ldots. \leq X_n$
  - We need to avoid certain 7 permutations of $[X_1, X_2, \ldots., X_n]$.
  - These permutations are difficult to define in the current model.

A Dual Model

- Variables and Domains
  - Represent the board with n x n Boolean variables $B_{ij} \in [0..1]$.
- Attacking Constraints
  - $\sum B_{ij} = 1$ on all rows and columns, $\sum B_{ij} \leq 1$ on all diagonals.
- Symmetry Breaking Constraints
  - Flatten the 2-d matrix to a single sequence of variables.
    - E.g., append every row to the end of the first row.
  - Every symmetric configuration corresponds to a variable permutation of the original solution, which is easy to define.
  - Impose an order between the original solution and all the solutions obtained by the 7 permutations:
    - lex≤(B, π(B)) for all π.

Lexicographic Ordering Constraint

- Requires a sequence of variables to be lexicographically less than or equal to another sequence of variables.
- lex≤($[Y_1, Y_2, \ldots, Y_k], [Z_1, Z_2, \ldots, Z_k]$) holds iff:

  $Y_1 \leq Z_1$ AND

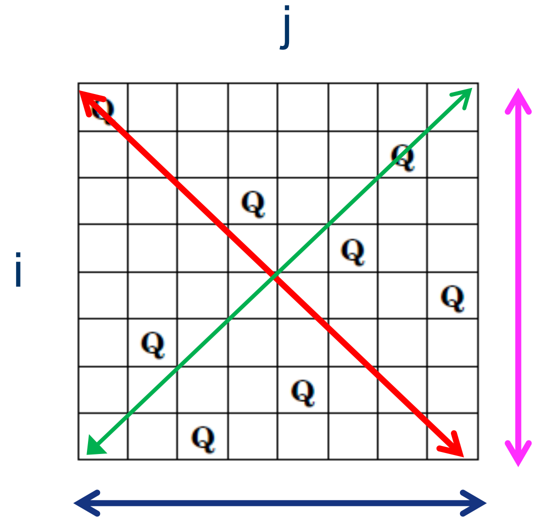
  $(Y_1 = Z_1 \rightarrow Y_2 \leq Z_2)$ AND

  $(Y_1 = Z_1$ AND $Y_2 = Z_2 \rightarrow Y_3 \leq Z_3) \ldots$

  $(Y_1 = Z_1$ AND $Y_2 = Z_2 \ldots. Y_{k-1} = Z_{k-1} \rightarrow Y_k \leq Z_k)$

  - lex≤([1, 2, 4],[1, 3, 3])

Symmetry Breaking in N-Queens



- lex≤(B, [B_{ji} | i, j ∈ [1..n]])
- lex≤(B, [B_{ij} | i ∈ [n..1], j ∈ [1..n]])
- lex≤(B, [B_{ji} | i, j ∈ [n..1]])
- lex≤(B, [B_{ij} | i ∈ [1..n], j ∈ [n..1]])
- …

- i, j → j,i
- i,j → reverse i, j
- i,j → reverse j, reverse i
- i,j → i, reverse j
- …

Symmetry Breaking in N-Queens

- lex≤(B, [B_{ji} | i, j ∊ [1..n]])
- lex≤(B, [B_{ij} | i ∊ [n..1], j ∊ [1..n]])
- lex≤(B, [B_{ji} | i ∊ [1..n], j ∊ [n..1]])
- lex≤(B, [B_{ij} | i ∊ [1..n], j ∊ [n..1]])
- lex≤(B, [B_{ji} | i ∊ [n..1], j ∊ [1..n]])
- lex≤(B, [B_{ij} | i, j ∊ [n..1]])
- lex≤(B, [B_{ji} | i, j ∊ [n..1]])

Which Model?

- Alldiff Model
  - $[X_1, X_2, \ldots, X_n] \in [1..n]$
  - alldifferent($[X_1, X_2, \ldots, X_n]$)
  - alldifferent($[X_1 + 1, X_2 + 2, \ldots, X_n + n]$)
  - alldifferent($[X_1 - 1, X_2 - 2, \ldots, X_n - n]$)

☺ Global constraints
☹ No easy symmetry breaking

- Boolean Symmetry Breaking Model
  - n x n $B_{ij} \in [0..1]$
  - $\sum B_{ij} = 1$ on all rows, columns
  - $\sum B_{ij} \leq 1$ on diagonals
  - lex≤(B , π(B)) for all π

☺ Easy symmetry breaking
☹ No global constraints

Which Model?

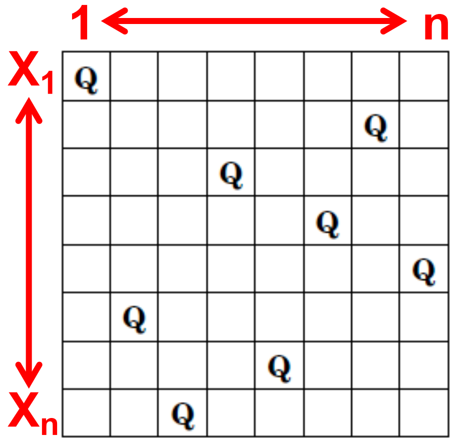
- Combined model
  - If you can't beat them, combine them ☺
  - Keep both models and use channeling constraints to maintain consistency between the variables of the two models.
  - Benefits:
    - Facilitation of the expression of constraints.
    - Enhanced constraint propagation.
    - More options for search variables.

A Combined Model

- Variables
  - for all i, $X_i \in [1..n]$, for all i, j $B_{ij} \in [0..1]$
- Constraints
  - alldifferent($[X_1, X_2, \ldots, X_n]$)
  - alldifferent($[X_1 + 1, X_2 + 2, \ldots, X_n + n]$)
  - alldifferent($[X_1 - 1, X_2 - 2, \ldots, X_n - n]$)
  - lex≤(B , π(B)) for all π
- Channeling Constraints
  - for all i,j $X_i = j \leftrightarrow B_{ij} = 1$

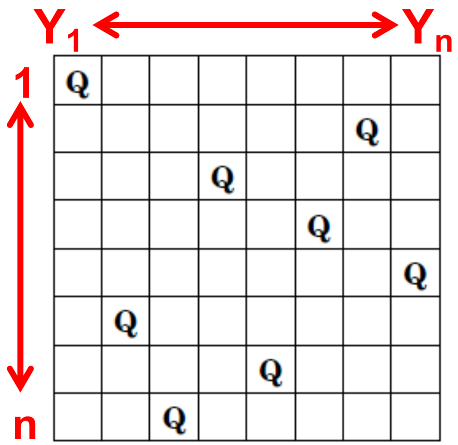
Do you notice something?

Dual Model of the Original Model?



- Variables and Domains
  - A variable for each row $[X_1, X_2, \ldots, X_n]$ → no row attack
  - Domain values [1..n] represent the columns:
    - $X_i = j$ means that the queen in row i is in column j
- Constraints
  - alldifferent($[X_1, X_2, \ldots, X_n]$) → no column attack
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$ → no diagonal attack

Another Dual Model



Both viewpoints yield the same CSP!

- Variables and Domains
  - A variable for each column $[Y_1, Y_2, \ldots, Y_n]$ → no column attack
  - Domain values [1..n] represent the rows:
    - $Y_i = j$ means that the queen in column i is in row j
- Constraints
  - alldifferent($[Y_1, Y_2, \ldots, Y_n]$) → no row attack
  - for all $i<j$ $|Y_i - Y_j| \neq |i - j|$ → no diagonal attack

Another Combined Model

- Variables
  - $[X_1, X_2, \ldots, X_n], [Y_1, Y_2, \ldots, Y_n] \in [1..n]$
- Constraints
  - alldifferent($[X_1, X_2, \ldots, X_n]$)
  - alldifferent($[Y_1, Y_2, \ldots, Y_n]$)
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$
  - for all $i<j$ $|Y_i - Y_j| \neq |i - j|$
- Channeling Constraints
  - for all i,j $X_i = j \leftrightarrow Y_j = i$

Is it really useful?

Another Combined Model

- Variables
  - $[X_1, X_2, \ldots, X_n], [Y_1, Y_2, \ldots, Y_n] \in [1..n]$
- Constraints
  - ~~alldifferent($[X_1, X_2, \ldots, X_n]$)~~
  - ~~alldifferent($[Y_1, Y_2, \ldots, Y_n]$)~~
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$
  - for all $i<j$ $|Y_i - Y_j| \neq |i - j|$
- Channeling Constraints
  - for all i,j $X_i = j \leftrightarrow Y_j = i$

What about this?

Another Combined Model

- Variables
  - $[X_1, X_2, \ldots, X_n], [Y_1, Y_2, \ldots, Y_n] \in [1..n]$
- Constraints
  - ~~alldifferent($[X_1, X_2, \ldots, X_n]$)~~
  - ~~alldifferent($[Y_1, Y_2, \ldots, Y_n]$)~~
  - for all $i<j$ $|X_i - X_j| \neq |i - j|$
  - ~~for all $i<j$ $|Y_i - Y_j| \neq |i - j|$~~
- Channeling Constraints
  - for all i,j $X_i = j \leftrightarrow Y_j = i$

and this?