Programming Language Theory
ICS313

Nancy E. Reed
nreed@hawaii.edu

Example problems
- N-queens
- cryptarithmatic

Solution strategies
- naïve (exponential)
- ......
- better $O(n)$, $O(n \cdot \log(n))$, or $O(n^2)$
- $O(1)$

Solutions
- Prolog
- Lisp

Problem Complexity
- Naïve solution – 8 queens, each in one of 64 squares = $64^8 = 281,474,976,710,656$ (over 281 trillion) (exponential/exponential)
- But, queens can’t be in the same square so there are 8 choose 64 = $64! / 56! \cdot 8! = 4,426,165,368$ (now only over 4 billion!) (factorial)
- But, the row numbers are a permutation of 1-8, and column numbers are the same, this helps a little, $8! \cdot 8! = 40,320 \cdot 40,320 = 1,625,702,400$ (factorial)

Problem Complexity (cont)
- However, we can let $q_n$ be the queen in column $n$, then the only thing left is to pick the row for each queen.
- The number of permutations of the integers 1 through 8 is $8! = 40,320$
- This is down greatly from over 4 trillion, but this is still factorial in the number of queens.

Eight Queens Problem
- On an 8-by-8 chessboard, place 8 queens so that they do not threaten each other. Queens are the most powerful piece and can move any number of squares in any direction along a row, column, or a diagonal
- Generalize to N queens problem where $N >= 4$ (why won’t it work for $N < 4$?)

How to Tell if Queens are on the Same Diagonal?
- Two queens are on the same upper left to bottom right diagonal iff the sum of the row and the column is the same for each
- Two queens are on the same upper right to bottom left diagonal iff the difference between the row and the column are the same for each

Prolog Solution to the 8 Queens problem
- http://www2.hawaii.edu/~nreed/ics313/assignments/prolog/8queens.pro
Prolog Solution to 8 Queens

solve(P) :- permutation([1, 2, 3, 4, 5, 6, 7, 8], P),
all_diff(S), % P is a solution,
all_diff(D). % the numbers 1-8

permutation([], []).
permutation([X | Y], Z) :-
permutation(Y, W), remove(X, Z, W).

remove(X, [X | R], R). % separate first part of list
remove(X, [F | R], [F | S]) :- remove(X, R, S).

combine([], [], [], []). % split lists
combine([X1 | X], [Y1 | Y], [S1 | S], [D1 | D]) :-
S1 is X1 + Y1, D1 is X1 - Y1,
combine(X, Y, S, D).

all_diff([X]),
all_diff([X | Y]) :- \=member(X, Y).

Lisp for the N-Queens Problem

- This polynomial time algorithm finds one solution!
- Note: printing the board is still \( n^2 \)
- A Lisp implementation is here:
  - N-Queens.lisp
- For more, see Marty Hall’s pages
  - http://www.apl.jhu.edu/~hall/NQueens.html

Cryptarithmetic Problem

- Find the unique digit that will replace each character to solve this puzzle
  
  \[
  \begin{array}{c}
  S E N D \\
  + M O R E \\
  \hline
  M O N E Y
  \end{array}
  \]

Representing Numbers with functor digit

\[
\begin{align*}
\text{digit}(0). \\
\text{digit}(1). \\
\text{digit}(2). \\
\text{digit}(3). \\
\text{digit}(4). \\
\text{digit}(5). \\
\text{digit}(6). \\
\text{digit}(7). \\
\text{digit}(8). \\
\text{digit}(9). \\
\text{carry}(0). \quad \text{carry}(1). \\
\text{\% carry digit must be 0 or 1} \\
\text{\% var = list of vars} \\
\text{DigitList=[S,E,N,D,M,O,R,Y],}
\end{align*}
\]

Representing Addition

\[
\begin{align*}
\text{\% carry digit must be 0 or 1} \\
\text{\% var = list of vars} \\
\text{DigitList=[S,E,N,D,M,O,R,Y],}
\end{align*}
\]