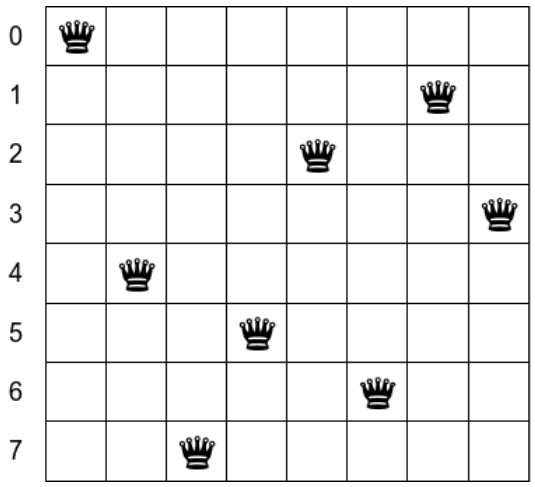
The n-Queens Problem with Unified Notations

Rhys Goldstein

December 14, 2008

Presented here is an algorithm that, given n, finds all solutions to the famous n-queens problem. Notably, this is done without code. Four small mathematical definitions, written using unified notations, serve as both a formal description and an implementation of the algorithm.

The n-queens problem requires one to place n queens on an n-by-n grid such that no two queens share a row, column, or diagonal. Below is one of many solutions with $n = 8$.



Taking advantage of the fact that all solutions have exactly one queen in each column, we represent each solution as a vector of n row numbers. The solution above is represented by the vector $[0, 4, 7, 5, 2, 6, 1, 3]$, as the first queen from the left is in row 0, the second is in

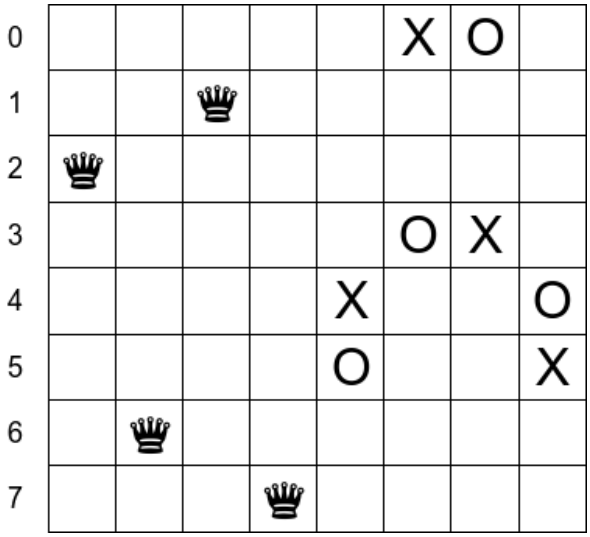
row 4, etc. We now wish to define a function named $queens$ that takes n as an argument, and results in a vector (of vectors) that represents all possible solutions to the n-queens problem. Note, for example, that if $n = 6$, there are 4 solutions. The equation below gives the result of $queens\,(6)$.

$$queens\,(6) \equiv \begin{bmatrix} [1,3,5,0,2,4] \\ [2,5,1,4,0,3] \\ [3,0,4,1,5,2] \\ [4,2,0,5,3,1] \end{bmatrix}$$

We may define $queens$ using another function named $complete$, which finds all solutions to the n-queens problem under the constraint that the leftmost queens have certain fixed positions. With an argument value of $[\,]$, none of the queens are fixed.

$$queens\,(n) := complete\,([\,])$$

Observe the diagram below, noting that $n = 8$.



In this example the leftmost four queens are fixed, and their positions are represented by the vector $[2, 6, 1, 7]$. Under this constraint, there are two possible solutions: one completed

by placing queens on the X's in the remaining columns, and the other completed using the O's. The function $complete$, given $[2, 6, 1, 7]$ as an argument, yields both solutions as shown below.

$$complete\,([2, 6, 1, 7]) \equiv \left[\begin{array}{l} [2, 6, 1, 7, 4, 0, 3, 5] \\ [2, 6, 1, 7, 5, 3, 0, 4] \end{array} \right]$$

To define $complete$ in general, we let the argument Q be the vector of queens on the left with fixed positions. We know that no two queens in Q share a column, and assume that there are no shared rows or diagonals either. This assumption holds true at least for the initial value ($Q \equiv [\,]$), which appeared in the definition of $queens$.

There are now two cases to consider. If the length of the vector Q is n, then all n queens have fixed positions. The result of $complete$ is then a vector containing the sole solution, which is Q. As for the other case, in which the length of Q is less then n, we use a new function named $check$. If invoked with the expression $check\,(i)$, this function fixes a queen at row i in the leftmost of the remaining columns, and yields all solutions under this tighter contraint. We invoke the function with the expression $check \circ ..n$, however, which applies $check$ to every row in the leftmost remaining column. The operator $||$ merely concatenates the results.

$\ll queens \gg$

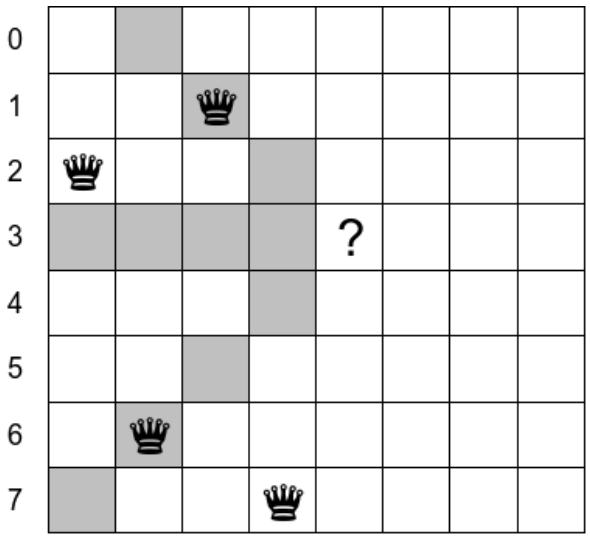
$$complete\,(Q) := \left(\begin{array}{ll} \#Q = n & \rightarrow [Q] \\ \#Q < n & \rightarrow ||\,(check \circ ..n) \end{array} \right)$$

Now we must define $check$ as a function of the row number i. It is tempting to simply add this new queen to the list of queens with fixed positions. This could be achieved by appending i to Q, and the result ($Q\,||\,[i]$) would then serve as a new argument to the function $complete$. We must remember, however, that when defining $complete$ we made the assumption that there were no shared rows or diagonals in Q. In order to maintain this property, we must consider the case in which there is a conflict between the new queen and any of the existing ones. This case, indicated by the boolean $conflict$, means that there are definitely no valid solutions to the n-queens problem contrained by $Q\,||\,[i]$.

$\ll \ldots ; complete \gg$

$$check\,(i) := \begin{pmatrix} conflict & \rightarrow [\,] \\ -conflict & \rightarrow complete\,(Q \,||\, [i]) \end{pmatrix}$$

Below we have $n = 8$, $Q \equiv [2, 6, 1, 7]$, and $i = 3$. To determine the value of $conflict$, we need need to check whether we may place in a queen on the square marked by the question mark. This square is in row i of the leftmost remaining column. In this case we may not place a queen there, as two of the existing queens share a diagonal with that square.



In general, if any queen in Q shares a row with the new queen, then the boolean vector $Q = i$ has at least one truthful value. If any existing queen shares a diagonal with the new one, then $|Q - i| = (\#Q - ..\#Q)$ has at least one truthful value. Combining and reducing these vectors, we obtain the following definition of $conflict$.

$\ll \ldots ; check \gg$

$$conflict := \vee\,((Q = i) \vee (|Q - i| = (\#Q - ..\#Q)))$$

This completes the implementation of an algorithm that finds all solutions to the n-queens problem. Despite the absence of code, it is fair to refer to the mathematical formulas as an "implementation", for with the aid of an intepreter or compiler the formulas could be

evaluated by a computer. A user could input $queens\,(8)$, and obtain as an output all 92 solutions with $n = 8$.

One advantage to using unified notations instead of code is the fact that one can manipulate parts of the implementation without introducing a new set of notations. As a demonstration, we will evaluate $conflict$ manually using the values of n, Q, and i given earlier.

$$\begin{aligned}
&conflict\ldots\\
&\quad = \vee\left((Q = i) \vee (|Q - i| = (\#Q - ..\#Q))\right)\ldots\\
&\quad = \vee\left(([2,6,1,7] = 3) \vee (|\,[2,6,1,7] - 3| = (4 - ..4))\right)\ldots\\
&\quad = \vee\left([\bot,\bot,\bot,\bot] \vee (|\,[-1,3,-2,4]\,| = (4 - [0,1,2,3]))\right)\ldots\\
&\quad = \vee\left([\bot,\bot,\bot,\bot] \vee ([1,3,2,4] = [4,3,2,1])\right)\ldots\\
&\quad = \vee\left([\bot,\bot,\bot,\bot] \vee [\bot,\top,\top,\bot]\right)\ldots\\
&\quad = \vee\,[\bot,\top,\top,\bot]\ldots\\
&\quad = \top
\end{aligned}$$

One might imagine that the use of code would yield a more compact implementation. But this mathematical version is surprisingly compact itself, as one observes once the surrounding text and images are omitted.

$$queens\,(n) := complete\,([\,])$$

$$complete\,(Q) := \begin{pmatrix} \#Q = n & \rightarrow [Q] \\ \#Q < n & \rightarrow ||\,(check \circ ..n) \end{pmatrix}$$

$$check\,(i) := \begin{pmatrix} conflict & \rightarrow [\,] \\ -conflict & \rightarrow complete\,(Q\,||\,[i]) \end{pmatrix}$$

$$conflict := \vee\left((Q = i) \vee (|Q - i| = (\#Q - ..\#Q))\right)$$