Local Search for CSPs

School of Electronic and Computer Engineering
Peking University

Wang Wenmin

Contents

Local Search for CSPs CSPs的局部搜索

- Local search algorithms are also effective in solving many CSPs, using complete-state formulation:

  局部搜索算法在求解许多CSPs问题中也很有效，采用完整状态形式化：

  - Initial state: assign a value for each variable

    初始状态：为每个变量赋值
  - Search: change the value of one variable at a time

    搜索：一次改变一个变量的值

Example: 8-queens problem

- Initial state is a random configuration of 8 queens in 8 columns, and each step moves a single queen to a new position in its column.

  初始状态是在8列上随机地摆放8个皇后，然后每一步将一个皇后移动到该列上的新位置。
- Typically, the initial guess violates several constraints, point is to eliminate the violated constraints.

  通常，初始状态会违反若干约束。要点是消除这些违反的约束。

Min-conflicts Heuristic 最小冲突启发式

- A most obvious heuristic is to select a new value for a variable.
  一个最明显的启发式是对一个变量选择一个新的值。
- Features 特点
  - Variable selection: randomly select any conflicted variable
    变量选择：随机选择任意一个冲突变量
  - Value selection: select new value that results in a minimum number of conflicts with others
    值的选择：选择导致与其它变量呈现最少冲突的新值
- Surprisingly effective for many CSPs 对许多CSPs出奇的有效
  - on *n*-queens problem, the run time of min-conflicts is roughly *independent of problem size*.
    在*n*皇后问题上，最小冲突的运行时间与问题大小基本上无关。
  - even on *million*-queens problem, it solves in an average of 50 steps, after the initial assignment!
    甚至对于百万皇后问题，在初始赋值后，平均50步左右就能得到解！

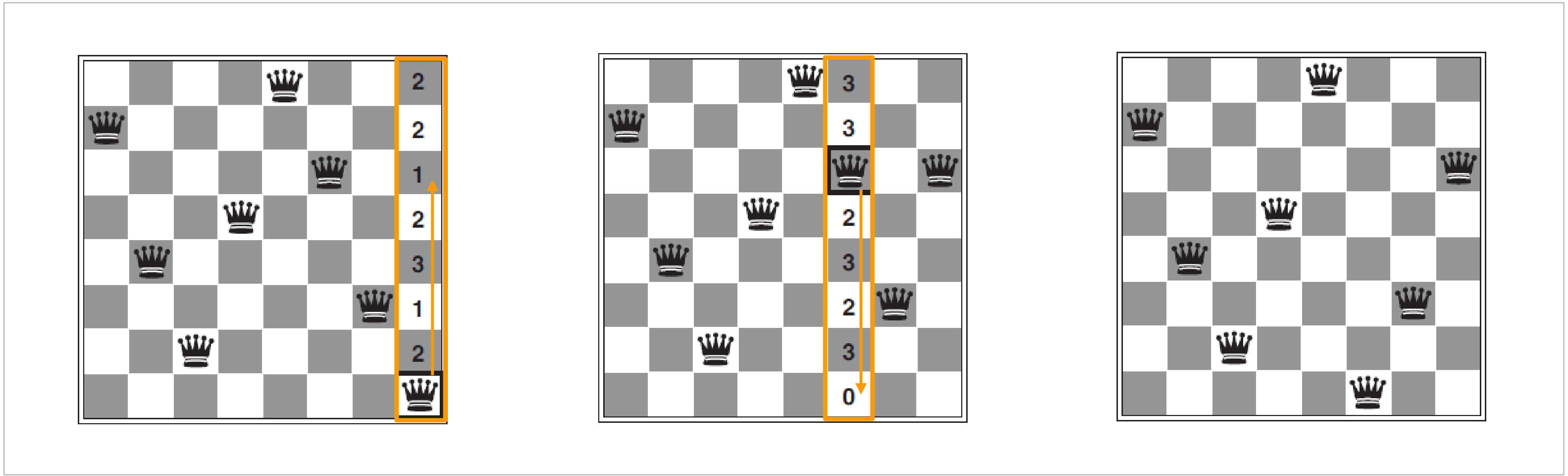
Algorithm Solving CSPs by Local Search 局部搜索求解CSPs算法

function MIN-CONFLICTS(*csp*, *max_steps*) **returns** a solution or failure
  inputs: *csp*, a constraint satisfaction problem
      max_steps, the number of steps allowed before giving up
  current ← an initial complete assignment for *csp*
  for *i* = 1 to *max_steps* **do**
    if *current* is a solution for *csp* **then return** *current*
    var ← a randomly chosen conflicted variable from *csp*.VARIABLES
    value ← the value *v* for *var* that minimizes CONFLICTS(*var*, *v*, *current*, *csp*)
    set *var* = *value* in *current*
  return *failure*

The initial state may be chosen randomly that chooses a min-conflict value for each variable in turn.
The CONFLICTS function counts the number of constraints violated by a particular value, given the rest of the current assignment.

初始状态可以随机选择，依次为每个变量选择一个最小冲突值。
给定当前赋值的其余部分，CONFLICTS函数计算特定值违反约束的个数。

Example: Min-conflicts for an 8-Queens Problem 8皇后问题的最小冲突法



A two-step solution using min-conflicts 用最小冲突法的两部解决方案

At each stage, a queen is chosen for reassignment in its column. The number of attacking queens is shown in each square. The algorithm moves the queen to the min-conflicts square, breaking ties randomly.

每一阶段，选择一个皇后在该列上重新分配。每个方格显示皇后攻击的数量。
算法将皇后移到最小冲突的方格内，随机地切断束缚。

Constraint Weighting 约束加权

- Can focus on the important constraints 能聚焦于重要的约束
  - Give each constraint a numeric weight, W_i, initially all 1.
    给定每个约束一个数值权重，W_i，初始值都为1。
  - At each step, choose a variable/value pair to change, that will result in lowest total weight of all violated constraints.
    每一步，选择一个变量/值对加以改变，这将导致所有违反约束的总权重为最低。
  - Then, weights are adjusted by incrementing the weight of each constraint that is violated by current assignment.
    然后，通过增加当前分配所违反的每个约束的权重来调整权重。
- Two benefits 两个益处
  - Add topography to plateau, making sure that it is possible to improve from the current state
    对高原增加了地势，确保能够改善当前的状态。
  - Also add weight to the constraints that are proving difficult to solve.
    对证明难以求解的约束也增加权重。

Thank you for your attention!