AI algorithms: Constraint Backtracking

(Chronological) Backtracking
Backjumping
Backmarking
Intelligent Backtracking
Dynamic Search Rearrangement

Input:
A constraint network with n variables v_i and constraints $c(v_i, v_j)$
A set of possible assignments $a_{i,k}$ for each variable v_i
A variable *jumpback* (init 1) and array *checkdepth*
A matrix *checkdepth* and array *backup*
A set *NG* of no-goods

Output:
An assignment $a_{i,k}$ for each variable v_i where all constraints $c(v_i, v_j)$ are respected

```
Algorithm ($depth, jumpback, checkdepth, backup$):
  for all $a_{depth,k}$ (ordered according to a fail-first heuristic) do
    if $checkdepth_{depth,k} \geq backup_{depth}$ then
      $v_{depth} \leftarrow a_{depth,k}$
      if current assignment is covered by a no-good then
        return (backtrack)
      end if
      $checkdepth_{depth,k} \leftarrow 1$
      for all $c(v_i, v_{depth})$, $backup_{depth} \leq i < depth$ do
        if $c(v_i, v_{depth})$ holds then
          $checkdepth_{depth,k} \leftarrow checkdepth_{depth,k} + 1$
        end if
      end for
      if $checkdepth_{depth,k} = depth$ then
        if $depth = n$ then
          return solution $(v_1,...,v_n)$
        end if
        recurse($depth + 1, jumpback, checkdepth, backup$)
        if $jumpback < depth$ then
          return (backtrack)
        end if
      end if
    end if
  end for
  $jumpback \leftarrow max_k(checkdepth_k)$
  for $depth \leq i \leq n$ do
    $backup_i \leftarrow depth - 1$
  end for
  if no assignment $a_{depth,k}$ was possible then
    Deduce no-goods and add to $NG$
  end if
```