Programmation logique en Oz

Le problème des n dames est : placer n dames sur un échiquier de dimension $n \times n$ tel que les dames ne se menacent pas (ni sur une ligne, ni sur une colonne, ni sur une des deux diagonales).

On vois immédiatement que toute solution doit placer les dames sur des lignes différentes. On peut donc représenter toute solution par une fonction qui envoie une valeur entre 1 et n (la ligne sur laquelle une dame est placée) vers une valeur entre 1 et n (la colonne sur laquelle la même dame est placée). Évidemment, les colonnes doivent être différentes. Donc, toute solution candidate est une permutation de la liste $[1 \ \ldots \ n]$, et elle est même une solution quand il n'y a pas menace sur les deux diagonales.

Exercice 1

1. Écrire une fonction {`Range N`} qui donne la liste des nombres de 1 à N. Il s'agit d'une fonction classique (pas de `choix`).

2. Écrire une procédure {`Select L X R`} qui, quand L est une liste, réussi quand X est un élément quelconque de L, et R est L sans X. Utiliser des `choix`.

3. Écrire une procédure {`Permutations L P`} qui réussi quand P est une permutation de la liste L. Utiliser des `choix`. On voudrait pouvoir obtenir *toutes* les permutation d'une liste L à l'aide de `SearchAll` :

   ```
   {SearchAll fun {$} {Permutations L} end}
   ```

4. Écrire une fonction {`Queens N`} qui donne toutes les solutions du problème de N dames. Utiliser dans un premier temps une approche *générer et tester* : générer toutes les solutions candidates, et tester s'il s'agit d'une solution.

Exercice 2

Écrire une fonction qui prend un argument N et qui donne toutes les solutions du problème des N dames, mais qui est plus efficace que celle de l'exercice 1. L'idée est de travailler avec des solutions partielles qui contiennent des placements seulement pour quelques unes des dames. Une solution partielle est étendue par le placement d'une autre dame seulement quand la solution partielle est cohérente.