Reasoning behind *ocamltsp*

Marek Mateusz Narozniak

June 6, 2016

1 Introduction

The purpose of *ocamltsp* is to demonstrate abilities of writing basic OCaml programs and share my knowledge of optimization. The purpose of this document is to show some formal logic behind it.

Writing a complete formal proof of *ocamltsp* is beyond the scope of that demo, however I would like to explain some reasoning behind it, that in future can serve to write a proper proof of optimality.

2 Optimal solution

2.1 Definitions

Let $path'$ be a path for problem instance P and p_{min} be a distance of $path'$. Let's assume $path'$ fulfills following properties

1 $path'$ is a Hamiltonian cycle

2 p_{min} is of globally minimal distance

Properties 1 and 2 define optimal solution to instance P of a *Traveling Salesman Problem*.

2.2 Program properties

For convenience main search function optimize is renamed to f.

$$f(path, d, ub) \text{ returns } (p, d')$$

Let's assume f be a function that accepts a partial path, it's distance d and best known *upper bound ub*. It returns a tuple of path p and distance d' associated to p.

Function f fulfills following properties

3 Final path will be a Hamiltonian cycle (source code line 4)

4 f performs a depth first search (recursive calls)

5 if branch is cut it implies that it doesn't contain solution better than best known solution (source code line 12)

6 optimal solution is guaranteed to be returned at the end

7 final solution is optimal solution to TSP

Some additional reasoning to justify points 5, 6 and 7.

(5) explored branch of $f(path, d, ub)$ will result with distance $d' \geq ub$

(6) suppose optimal solution p_{min} is not returned, this contradicts either property 4 not all solutions have been enumerated either property 5 optimal solution has been in one of excluded branches

(7) because $path'$ is a Hamiltonian cycle, if not then contradicts property 3 and p_{min} is of minimal overall length, if not contradicts property 6, if solution that is a Hamiltonian cycle of minimal overall distance is not a solution to *Traveling Salesman Problem* then it contradicts properties 1 and 2

3 Source code

```
let rec optimize(dom, dist, path, cost, ub, visit) =
  if dom = []
  then
    if hamiltonian(1, path, visit)
    then
      (cost, path)
    else
      (-1, [])
  else
    if length(hd(dom)) = 1
    then
      if cost + hd(hd(dist)) <= ub
      then
        optimize(tl(dom), tl(dist), insert(path, hd(hd(dom))),
             hd(hd(dist)) + cost, ub, visit)
      else
        (-1, [])
    else
      let (pub, ppath) = optimize(tl(dom), tl(dist), insert(
          path, hd(hd(dom))), hd(hd(dist)) + cost, ub, visit)
      in
        if pub != -1
        then
          if pub <= ub
          then
            let (qub, qpath) = optimize(tl(hd(dom)) :: tl(dom)
                , tl(hd(dist)) :: tl(dist), path, cost, pub,
                visit)
            in
              if qub != -1
              then
                if pub < qub
                then
                  (pub, ppath)
                else if qub < pub
                then
                  (qub, qpath)
                else
                  (qub, qpath)
              else
                (pub, ppath)
          else
            optimize(tl(hd(dom)) :: tl(dom), tl(hd(dist)) ::
                tl(dist), path, cost, ub, visit)
        else
          optimize(tl(hd(dom)) :: tl(dom), tl(hd(dist)) :: tl(
              dist), path, cost, ub, visit) ;;
```