Heuristic for “Go to Bucharest” that dominates SLD

- **Array A[i,j] = straight-line distance (SLD) from city i to city j; B = Bucharest;**
- **s(n) = successors of n;**
- **c(m,n) = {if (n in s(m)) then (one-step road distance m to n) else +infinity};**
- **s_k(n) = all descendants of n accessible from n in exactly k steps;**
- **S_k(n) = all descendants of n accessible from n in k steps or less;**
- **C_k(m,n)**

  = {if (n in S_k(m)) then (shortest road distance m to n in k steps or less)
  else +infinity};
- **s, c, are computable in O(b); s_k, S_k, C_k, are computable in O(b^k).**

- **These heuristics both dominate SLD, and h2 dominates h1:**
  - h1(n) = min_{x in Romania} (A[n,x] + A[x,B])
  - h2(n) = min_{x in s(n)} (c(n,x) + A[x,B])
- **This family of heuristics all dominate SLD, and i>j => h_i dominates h_ j:**
  - h_k(n) = min((min_{x in (S_k(n) ∩ S_k(B))} C_k(n,x)+C_k(x,B))),
    (min_{x in s_k(n), y in s_k(B)} (C_k(n,x) + A[x,y] + C_k(y,B)))
- **h_final(n) = same as bidirectional search; => exponential cost**