

Lecture 10 ESPRESSO

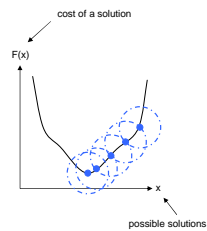
Some Problems are Hard

Using Exact Algorithms vs. Heuristics

- Quine-McCluskey
 - Calculated all prime implicants to derive the optimal solution(s)
 - Petrick's Method derives all covers to determine minimum cover set(s)
 - Number of prime implicants grow quickly -- solution space is huge!
 - Finding the minimum cover set in a class of NP complete problems
 - Determining optimal solution is difficult
- Move to heuristics
 - Look at generating a quality solution quickly (not necessarily optimal)

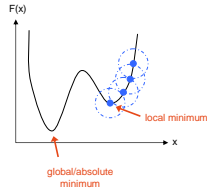
Local Search

- Don't generating all prime implicants and minterms
- Instead, ESPRESSO successively modify a given initial cover
 - This technique is called a *local search* algorithm
- Idea behind local search
 - Search space or solution space - set of all possible values and cost associated with solution
 - Start with an initial value
 - Search all points in neighborhood for a feasible point whose cost is less than current
 - Different problems have different neighborhood definitions
 - If one is found, start process over



Local Search

- Drawback of local searches is local optimality
 - Solution is locally optimal if its neighborhood does not contain any solutions with a lower cost
 - Locally optimal solution may not be the optimal solution
- Modify local search so we don't get stuck at the local minimum



ECE 474a/575a
Susan Lysdecky

4 of 15

Espresso

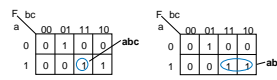
- Espresso utilizes local search (keeping in mind local minimum problem)
 - Probably most popular minimization algorithm
 - Extremely efficient Boolean manipulation
- Composed of three main operations
 - EXPAND, REDUCE, IRREDUNDANT
- Other operations include
 - COMPLEMENT, ESSENTIAL PRIMES, LASTGASP, MAKESPARSE
- Espresso Heuristic (in a nutshell)
 - Apply Expand and Irredundant operators to optimize the current function specification
 - Uses the reduce operator to get out of local minimum
 - Iterated until the solution converges

ECE 474a/575a
Susan Lysdecky

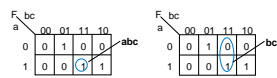
5 of 15

Espresso – Expand Operator Overview

- EXPAND
 - Deleting one (or more) of its literals
 - Check for validity



Expand abc by removing c (results in ab)
Is it valid? Yes.



Expand abc by removing a (results in bc)
Is it valid? No.

ECE 474a/575a
Susan Lysdecky

6 of 15

Espresso

```
espresso(F,D) {  
    R = complement(F U D);           ← F is the on-set, D is the don't care set  
    F = expand(F,R);                 // initial expansion  
    F = irredundant(F,D);           // initial irredundant cover  
    E = essentials(F,D);           // detect essential prime implicants  
    F = F - E;                       // remove essential prime implicants from f  
    D = D U E;                       // add essential prime implicants to D  
    repeat {  
         $\phi_i = |F|$ ;  
        F = reduce(F,D);  
        F = expand(F,R);  
        F = irredundant(F,D);  
    } until (|F|  $\geq \phi_i$ );  
    F = F U E;  
    D = D - E;  
    RETURN F;  
}
```

repeated application of REDUCE, EXPAND, IRREDUNDANT operations while cost keeps decreasing

ECE 474a/575a
Susan Lysdecky

13 of 15

ESPRESSO, to be continued...

- We've seen the high-level idea behind ESPRESSO
 - ESPRESSO performs extremely efficient Boolean manipulation
- How are these operations actually performed?
- How is data represented?

ECE 474a/575a
Susan Lysdecky

14 of 15
