GBL rev n 6/20

Department of Electrical and Computer Engineering

**CSCE686 Advanced Algorithm Design**

Spring Quarter 2020

**Complex Optimization Algorithm Design Project**

**Project Problem Beginnings**

The intent of this individual project is to incorporate the major educational aspects of search algorithm development for combinatoric complex optimization problem solutions (NP-Complete, PSpace, …) . These discrete optimization projects can be an aspect of the student’s research or selected from the list provided (*note that each student should have a unique project*).

**General Educational Objectives:**

**1.** Develop an *ability to design and evaluate algorithms* for solving complex NPC or more complex scientific and engineering optimization problems using explicitly the design techniques presented in class. This includes explicit problem domain specifications and selection and integration of appropriate search algorithm templates from the deterministic and stochastic spectrum. This should result in software engineered code that is effective and efficient and well documented. And, thus, should make testing less complex and maintenance less expensive.

**2.** Develop an ability to define appropriate experimental design in order to characterize computational performance (*effectiveness and efficiency*) for solving such optimization problems.

**3**. Be able to represent results in a professional well written journal/conference format with *qualitative* conclusions supported by *quantitative* data.

**Project Development: *(Define your project problem!)***

1. **Problem Selection:** (English → symbols → math/logic).   
   Select and discuss a specific real-world discrete optimization problem. It should **incorporate at least TWO NP-Complete problem models (multi-objectuve).** Aggregate objectives into one evaluation function (additive, …). Another project selections include **a dynamic NPC problem with a nonlinear model or with extensive constraints or a PSPACE-Complete Problem**. What does the search landscape look like? Thoughts on testing.
2. **Problem DomainSpecification:** (20 pts)Define in set/sequence **formal structural notation** and then as a 0/1 symbolic structureproblem as appropriate. Discuss it from a number of points of view; i.e., objects, process, constraints (variable bounds),state-space, applications, goal.... Also, relate to a graphical data structure representation of the problem. What are Di and Do? Develop and show the associated problem domain order-of analysis(complexity). Include references. [English → symbols → math/logic]
3. **Algorithm Domain Selection & Specification**: (20 pts)Select and analyze in detail various appropriate search algorithm constructs and template abstractions (deterministic, stochastic) to solve the specific problem. 1) Employ at least *one* ***deterministic* *search technique*** (global DFS.BT, global BFS, Z\*, A\*) possibly with an ID, beam or other restrictive search method 2) stochastic search (multi-population) and 3) local search (single population) . Use algorithm design template for top-down development using nomenclature of texts, as reflected in the course notes and HW.
4. **Algorithm (program) Design and Refinement (MAJOR EMPHASIS):** (60 pts)Develop in depth the given problem domain/algorithm domain integration design process for your problem. Reflect explicitly extensive design understanding via the various epochs of program refinement. Relate to the given implementation via *creative program design refinements* towards efficiency. Indicate the refinements of the data structures and ADTs: input variables, state variables, constant parameters, exogenous PD variables, random variables, output variables and creative selection of heuristics. Discuss and show program refinement to commented pseudo code (*use standard search form* –pseudocode.sty). Design should include a number of design refinement steps. Include references as appropriate. Address the question of program correctness (behavior) leading to implementation along with algorithm complexity. Address *approximation search techniques* in other algorithm designs*.*
5. **Implementation: (NOT REQUIRED!)** Design and use (select, implement or use existing software) a software system using object-oriented/functional design to solve the specific problem. Documentation should be in accordance with **standards** (AFIT/ENG, C, C++, Java, ...) Including an object-oriented or functional design analysis, design and implementation. Include structure charts and ADTs as well as algorithm order-of analysis (complexity) in commented code. Use existing code (reusable modules) as appropriate (please)! Evaluate used code via software engineering principles.
6. **Design of Experiments and Testing**: (NOT REQUIRED!) Define the **objectives of the experiments**. Use appropriate computational experimentation methods (objectives, statistical comparison, …). Solve for at least three examples; one of low dimension for pedagogical reasons, a medium size example, and another with large dimension to reflect general applicability of search designs. For your particular problem domain, suggest use of a library of example problems or benchmarks if they exist. Develop an algorithmic and graphical representation of search process (e.g. deterministic graph search tree) for your problem.
7. **Extended Development:** Relate the specific NPC problem to *at least* 1 other classical NP-Complete problem through adhoc mapping descriptions; that is, prove your problem is NPC via data structure mapping in polynomial time (both ways). If your problem is worst than NPC, show mapping to complete problems in that problem space.
8. **Report:** Incorporate all aspects of project into **final formal report**. Follow suggested outline, software engineering principles and use Latex or Word in a two-column IEEE or ACM paper format (available on the web). Be concise in your presentation and follow Barr’s, MF’s, and Talbi’s suggestions. In your report relate to the algorithmic design process, discuss specifically your generic algorithm design approaches as explicitly defined in class and in other references.

**Reporting Dates at least by**

**(***Graded Documentation - incomplete informal reports with REFs per following dates)*

**- June 12:** InitialProject Selection by student, initial English and some math notation

**-** **June 12:** Problem Description – English and math notation with CSCE686 set/sequence notation  
 and thoughts on experimental examples. (20 pts)

**- June 15:** ProblemSelection Approval & Assignment by Professor

**- June 19**: Problem Domain Specification and Selection of **Deterministic** Search Algorithm

**- June 19**: Algorithm Design Deterministic approach – dfs or bfs, high to low level algorithm   
 refinement – pseudo code – *approximation search techniques* (with search elements)

(40 pts)

**- June 26**: Problem Domain Specification and Selection of **Stochastic** Search Algorithm

**- June 26**: Algorithm Design Stochastic approach – high to low level algorithm   
 refinement – pseudo code – with search elements - maybe with local search?  
 (40 pts)

- **July 3:** Final Project Report - thoughts on computation and experimental design and based   
 upon your algorithmic design feedback from Professor. (incorporating the design)