**Michael Yahner  
Matthew Farkas**

**Optimization for Parallel Compilers : Project Part 1**

**Use Instructions**

Starting the program: The project files are submitted together in a .zip file. Download the zip file to a directory on your computer and unzip the file. Navigate to and start the program located at “OptimizingParallelCompiler\bin\Debug\OptimizingParallelCompiler.exe”.

Using the program: The File menu located in the upper left hand corner contains a list of items that can be clicked to run the following functions. Note, you should also run them in the following order to produce desired results. 1) “Load Oneil Code” – Use this menu item to load test programs. Clicking on a test program will load the code into the leftmost textbox. Text changes can also be made in this textbox to modify, or run your own ad-hoc “Oneil” code. 2) “Convert” – will convert “Oneil” code into C# code into the textbox on the right-site. 3) “Compile” will compile the C# code on the right into a runable executable written to the bin folder. 4) “Run” – will run the created executable. 5) “Exit – exits the program.

Shortcuts: As an alternative to the “File” menu, the following key combo shortcuts can be used:

Convert: Control + Shift + C  
Compile: Control + C  
Run: Control + Shift + R  
Exit: Alt + R

**Overview**

**Tests**

Test programs served as our test cases.

**Unexpected behavior**

Automation: compile fails due to finish label having no statements.  
Jacobi: fails due to no multi dimensional arrays being implemented.  
Multiply: doesn’t not compile because it contains a keyword of the language as a goto and a label.  
Sort: is using a key word as goto and label.  
Sort Insertion: is using a key word as goto and label.  
Tax: contains unassigned variable usage.  
Trivia: when anything is 50 or above, program crashes with “divide by zero” error