ITP30002 Operating System

Homework 2

Parallel TSP Solver

Date assigned:
Apr 17 (Fri)

Due date:
9 PM, Apr 27 (Mon)

Overview

- You are asked to write `ptsp.c`, a multi-process program that solves a given Traveling Salesman Problem (TSP) instance
  - The main process spawns multiple children processes to explore different parts of the solution space concurrently
  - Construct the program according to the given functional and design requirements
  - Use process creation and join, and unnamed pipes, and signaling properly for implementing required functionalities of `ptsp.c`

- Submission
  - Deadline: 9 PM, April 27 (Mon)
  - Late submission: no late submission will be accepted
  - Deliverables: (1) write-up, and (2) source code
  - Submission site: Hisnet

Peace Server

- Recommend to use Peace http://peace.handong.edu
  - Ubuntu 16.04.6 LTS Kernel 4.15.0
  - GCC 5.4.0
  - Intel Xeon CPU 2.2 GHz 40 cores, 32 GB RAM

- Accounts
  - Continue to use your own account if you already have one
  - Find ID/PW at the homework repository in Hisnet
    - TAs are creating new accounts for those who declare they do not have an account at the April 3 survey.

- Note that your program should be built and executed successfully because TAs will test your program using Peace

Notes

- Help desk
  - TAs will offer help desk session next week. The schedule is TBA.
  - Feel free to ask if you have a trouble using Peace anytime

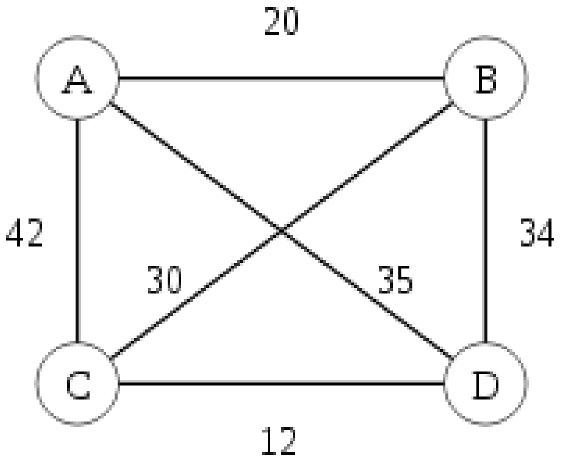
- Sample data & code: https://github.com/hongshin/OperatingSystem/tree/hw2
  - You can find sample data files (`gr17.tsp`, `gr21.tsp` , `gr24.tsp`, `gr48.tsp`); they are obtained from the Concorde TSP benchmark
  - You can also find `tsp17.c`, a sequential program for solving `gr17.tsp`. This is a just sample program for explaining how to solve TSP. No need to reuse this program for doing this homework.

- Note that we will revisit TSP for another homework of this course

Background: Travelling Salesman Problem (TSP)

- TSP is to find a shortest possible route in a given fully-connected undirected weighted graph
  - a route is a path starting from a node that visits all other nodes exactly once (no revisit) and then ends at the starting node
  - the length of a route is the sum of all weights of the toured edges

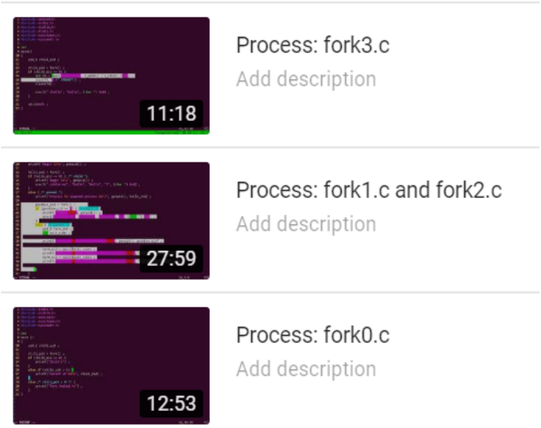
- TSP is NP-complete that we need to check all tours in order to find an optimal solution of a given problem



https://en.wikipedia.org/wiki/Travelling_salesman_problem

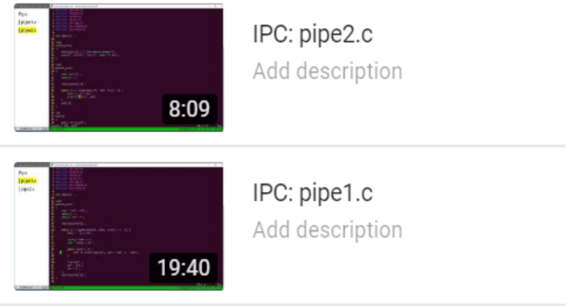
shorted route: <A, D, C, B, A>
- length: 97 (= 35 + 12 + 30 + 20)

Background: System Programming


11:18
Process: fork3.c
Add description

27:59
Process: fork1.c and fork2.c
Add description

12:53
Process: fork0.c
Add description


8:09
IPC: pipe2.c
Add description

19:40
IPC: pipe1.c
Add description

16:37
IPC: signal3.c and signal4.c
Add description

13:30
IPC: signal1.c and signal2.c
Add description

- Useful links
  - Linux man pages https://linux.die.net/man/
  - GNU C library https://www.gnu.org/software/libc/manual/
  - Advanced Linux programming http://www.makelinux.net/alp/
  - The Linux Programming Interface http://man7.org/tlpi/code/online/all_files_by_chapter.html
  - Unix Application and System Programming by Prof. Stewart Weiss http://www.compsci.hunter.cuny.edu/~sweiss/course_materials/unix_lecture_notes.php

Your Assignment

- Construct `ptsp.c` that receives a TSP instance and then finds a shortest route using concurrent executions of children processes

- Requirements
  - Your program must provide all expected functionalities
    - Input
    - Output
  - You must implement the program by following the given design and applying proper programming features/mechanisms
    - Main process behavior
    - Children process behavior

Functionalities (Requirement)

- Input
  - Receive (1) a filename of a TSP instance data, and (2) the limit of the number of children processes at a time as command-line arguments
    - E.g., `$ ./ptsp gr17.tsp 8`
  - A TSP data file with N cities (City 0 to City N-1) consists of N lines where each line contains N non-negative integers for $13 \leq N \leq 50$.
    The j-th integer at the i-th line is the weight between City i to City j
  - The limit of children processes at a time is in between 1 and 12 (inclusive)
- Output
  - When a user raises a termination signal (i.e., Ctrl+C) or all subtasks are done completely, print out the following information to the standard output and then terminates the program:
    - the best solution (a route and its length) upto the point
    - the total number of checked/covered routes upto the point

Program Design (Requirement)

- Main process
  - The main process should be single-threaded
  - Spawns a child process to delegate a subtask if there is a remaining subtask.
    - a subtask is specified by a prefix of routues
    - a subtask is to explore all routes (permutations) having a certain prefix
    - the number of routes to explore in a subtask should be 12!
    - each subtask is to cover a unique set of routes
(no overlapped with other subtasks)
  - Always runs as many children processes upto the given limit as possible at a time
  - Receives the result from a child process through an unnamed pipe
  - When the user raises a termination signal, prints a best solution among all checked routes including the ones by existing children process
  - Terminates all children processes (if there exists) before a termination

Program Design (Requirement)

- Child process
  - A child process should be single-threaded
  - A child process should not spawn another child process
  - Finds an optimal solution in the assigned subspace (subtask)
  - Terminates when the search for the assigned subtask is completed, or a termination signal is received
  - Transfers the best solution and the number checked routes to the main process via unnamed pipe when it terminates

Evaluation

- Criteria

- Fulfillment of requirements	50%
- Novelty of analysis and discussion	30%
- Clarity in technical description	20%

- Write-up
  - up to 3 pages in the given template
  - describe your understanding of the problem, and your solution toward it
  - analyze the performance of your program with respect to the number of employed children processes
  - show that your program fulfills the requirements
  - discuss issues and ideas as you had in doing this homework
  - submit a PDF file of your write-up