**AMS 550.400 HW SET 1 Due Date: Oct 8**

Last Compiled on October 10, 2012

---

**Problem 1 (10 pts):**

1. mkdir hw1p1.git
2. cd hw1p1.git
3. git init .
4. vi main.txt
5. git add .
6. git commit -m "A added to master"
7. vi main.txt
8. git add .
9. git commit -m "B added to master branch"
10. git branch alt
11. git branch
12. vi main.txt
13. git add .
14. git commit -m "C added to master branch"
15. git checkout alt
16. vi main.txt
17. git add .
18. git commit -m "X added to alt branch"
19. git checkout master
20. git merge alt
21. vi main.txt
22. git add .
23. git commit -m "Alt merged to master branch"

24. git branch
25. vi main.txt
26. git add .
27. git commit -m "D added to master branch"
28. git log –graph –oneline
29. git checkout alt
30. git log –graph –oneline

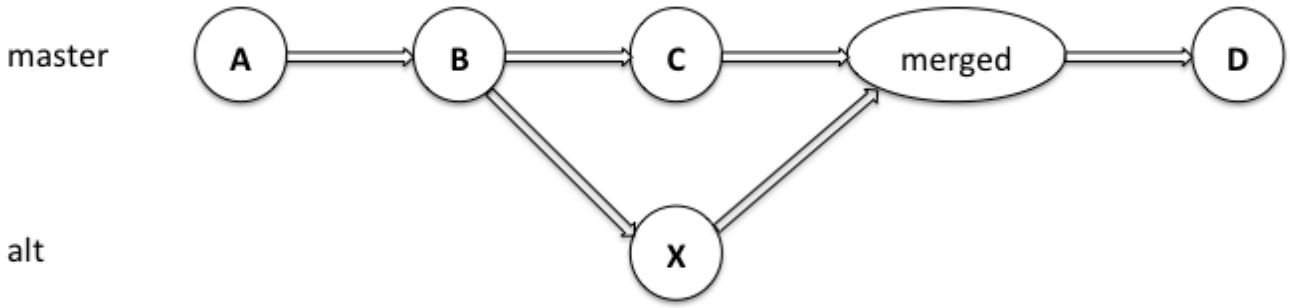


Figure 1: Master Commit Graph

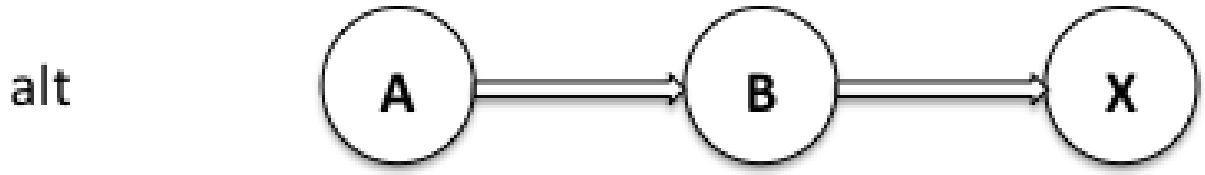


Figure 2: Alt Commit Graph

---

**Problem 2 (10 pts):**

1. mkdir hw1p2.git
2. cd hw1p2.git
3. git init .

4. git remote add s1 git://github.com/nhlee/550400.stanza1.git
5. git pull s1 master
6. vi main.txt
7. git add .
8. git commit -m "Title added"
9. git remote add s2 git://github.com/nhlee/550400.stanza2.git
10. git pull s2 master
11. vi main.txt
12. git add .
13. git commit -m "2nd stanza merged"
14. git remote add s3 git://github.com/nhlee/550400.stanza3.git
15. git pull s3 master
16. vi main.txt
17. git add .
18. git commit -m "3rd stanza merged"
19. git remote add origin https://github.com/tangdnn/550400.homeworkset.1.git
20. git push origin master
21. git remote rm origin

---

**Problem 3 (40 pts):**

For this exercise, we wish to build a cooperative strategy for a team of four students who have split up the presentation into four parts.

**Strategy 1:**

- **Formulate the Problem.** Since they do not wish to work concurrently, we must develop a work flow strategy for the team to merge all four parts together without the need for a group meeting using `git`.

- **Outline the Model.** One work flow strategy for this team is to for the team members to merge their presentation consecutively; i.e.: $B$ merge with $A$, then $B$ merge with $C$, and lastly, $D$ merge with $C$. The endogenous variable of this model is for all four team members to combine their respective parts of the presentation without working concurrently during a group meeting. The exogenous variable of this model is an effective work flow strategy proposal—in this case, merging each part consecutively using `git`—to combine different parts of the presentation. Unimportant variables that can be neglected are the amount of time it takes to write each part on its own (assuming that at least two parts are done at the time of merging), the physical quantity of each part, and the consistency of each students' writing. In this case, the combination of all four parts of the presentation (endogenous variable) depends on the process of consecutive merging work flow strategy (exogenous variable).

- **Is It Useful?** Yes, the consecutive merging work flow model is useful. The model fulfills all of the requirements set by the problem. Since model requires $B$, $C$, and $D$ to merge each previous part, the actual merging does not require any group members to be present, least of all at the same time. The merging can be done on team member's own time. More importantly, this strategy allows all four parts to be merged in `git`.

- **Test the Model.** We can predict that this model is efficient. Since the whole model can be done in very quickly, assuming that all students have their respective parts done at the time of merging, we can say that the model is efficient. We can also test that this model can be done quickly. Even if not all students have their parts done at the same time, each merging step would only take a minute or two. Thus, we can say that this model can also be done quickly.

- **Strengths and Weaknesses.** The advantages of this model is that it allows the team members to combine their parts of the presentation together without needing them to be present at the same time. The model is also quick and efficient, as explored in the Test the Model section. However, one major disadvantage is that since there is no requirement for all students to have their parts done at the same time, each merging step might be delayed since it depends on whether or not one of the students finished their part. The model might be held up if $A$ did not finish the *Introduction*, and students $B$, $C$, and $D$ have to wait for $A$ to finish in order to combine the presentation.

**Strategy 2:**

- **Formulate the Problem.** Like the previous model, the problem remains the same. We must develop a *different* work flow strategy for the team to merge all four parts together without the need for a group meeting using `git`.

- **Outline the Model.** Another work flow strategy for this team is for all team members to push their `git` folders to a shared `git` repository and then for one group member to pull the contents of that repo and merge all four parts manually. Like the previous strategy, the endogenous variable of this model is for all four team members to combine their respective parts of the presentation without working concurrently during a group meeting. The exogenous variable of this model is the proposed work flow strategy—in this case, pushing four `git` folders to the same `git` repo and for one group member to pull the contents and merge the four parts together. Unimportant variables that can be neglected are the amount of time it takes for

each students to write their part (since we are assuming that all parts are done at the time of merging), the physical quantity of each part, and the consistency of the students' writing. Like the previous strategy, the combination of all parts of the presentation (endogenous variable) depends on the simultaneous merging work flow strategy (exogenous variable).

- **Is It Useful?** Yes, the simultaneous merging work flow model is useful. The model fulfills all requirements set by the problem. Since the merging process is done by only one student, none of the team members need to meet at a group meeting to combine the presentation. Lastly, the model allows the presentation to be merged using `git`.
- **Test the Model.** To test this model, we can predict that the model is efficient. Assuming that the students have pushed their respective parts to the `git` repo at the time of merging, the whole model can be done soon after all parts are present in the repo. Thus, we can say that the model is efficient.
- **Strengths and Weaknesses.** The advantage of this model is that it allows team members to combine their parts of the presentation together without needing them to be present at the same time. This model is also efficient, as explored in the Test the Model section. One major disadvantage is similar to the previous model; the student doing the merging step might be waiting on all the other students to push their folder to the `git` repo, and it might be delayed if not all parts are present in the repo. Additionally, this model forces one group member to take on extra work for the project, which results in unfair distribution of work.

**Final recommendation:** We should use Strategy 1 for this presentation, since it has less disadvantages and more advantages than Strategy 2. It allows a more fair distribution of work and efficient process.

---

**Problem 4 (aka. Fair Play, 40 pts):**

To answer this question, we must first be familiar with the rules of tennis and to be able to define 'fair game.' A fair game is a game, whether athletic or academic, where all the contestants have the same chance of winning. In tennis, a fair tennis game is a game where both players have an equal chance of winning. However, one aspect of tennis seems to contradict the definition of 'fair game': One player will be serving the ball, which is assumed to be more advantageous than receiving the ball. We hope to explain this using a mathematical model.

- **Formulate the Problem.** In tennis, serving the ball is assumed to be more advantageous than receiving the ball, since players can use the serve to put more offensive power into the serve. Furthermore, the server has less chance to lose the set since they served the ball first. The problem we are looking to investigate is to determine if a tennis game is fair (i.e.: if serving the ball really does give the server an advantage over the receiver).
- **Outline the Model**
- **Is It Useful?**
- **Test the Model**