CSC 555 Social Computing

S-C Report

Name: Mohit Satarkar

Unity Id: mmsatar2

**(1) Goal Modeling**

Goals:

1. To be safe
2. To have pleasure
3. To have fame
4. To have privacy

 Plans:

1. To be safe = Share with common friends
2. To have pleasure = Share with companions
3. To have fame = Share with all
4. To have privacy = Share with none

**(2) Social Context Modeling**

Outline:

(location; location attributes + activity + companion; companion type)

Contexts:

(Graduation ceremony; Graduating; All)

(Library; Day time; Studying; Colleagues)

(Hurricane; Day time; Staying safe; Family)

(Airport; Night time; Travelling; Family, Friend, Colleague)

(Hiking; Exercise; Family, Friend)

(Presenting Paper at a Conference; Academics; Colleague)

(Bar with fake ID; Drinking; Friends)

**(3) Social Expectation Modeling**

Commitments:

Outline:

C​(subject; object; antecedent; consequent)

Example:

C(beast; friends; context = hiking a mountain; share = all)

Defined Commitments:

C(beast; {all}; context = Graduation Ceremony & companions = all; share = all)

C(beast; {all}; context = Library during the day & companions = all; share = none)

C(beast; {family, friends}; context = Hurricane during the day & companions = family, friends; share = common friends)

C(beast; {colleagues, family, friends}; context = Airport at night & companions = colleagues, family, friends; share = companions)

C(beast; {family, friends}; context = Hiking on a mountain & companions = family, friends; share = common friends)

C(beast; {colleagues}; context = Presenting a paper at a conference & companions = colleagues; share = companions)

C(beast; {all}; context = Bar with a fake ID & companions = all; share = none)

**Preferred tiers of sharing:**

1. Share with Common Friends
2. Share with Companions
3. Share with all
4. Share with none

**Algorithm:**

The data for the sanction recommendation is volatile. When you make checkins, and changes in the recommendation for sharing policy will be suggested based on feedback from tagged users. Once you stop running the code, and restart it, you’ll have initial data instead of modified data. So please test all code before stopping execution.

Basically, data is synced automatically by getting feedback once you run the first case. The next checkins are done by updating sharing policy using the feedbacks for checkin.

IF user provides negative feedback for checkin with share with none policy, we change share policy id 1 tier above it. If user provides negative feedback for checkin with share with all policy, we change share policy to 1 tier below it. We take votes for each user’s feedback and provide policy with most votes. If equality is found, we assign vote to our won preferred policy.

We have three main cases for the user to provide input in, which include:

1. Make a check in
2. Provide feedback to your tagger

These methods have the following algorithmic definitions:

1. Make a check in
   1. Get placeid and companions to tag
   2. Check these with the existing maps for your preferred sharing policy
   3. Send checkin based on the sharing policy
   4. Update the sharing policy based on accessed user feedbacks
2. Provide feedback to your tagger
   1. Get policy id preference in a map with key as place id.
   2. Get user groups preferred to be tagged with for a place in a map with key as place id
   3. Get list of unattended checkins and choose checkin id to respond to
   4. If policy Id and user group matches to our maps for that checkin from the user, provide positive sanction
   5. Else provide negative sanction