**CSE 571 Fall 2020 Team Project Description**

**General guideline**

* Team projects are open-minded—you have a lot more freedom than individual projects here; at the same time, this means more researching: more reading, exploration, potential for errors, and debugging of course. **Deadlines is Dec 7th. You have about a month to complete the project.**
* **You should have already received notifications about your project team assignment (3-4 students per project)**
* You may consult online resources. ***Plagiarism will be checked***.
* For your report, use the IEEE transactions template (Choose: Template Selector->Transactions->IEEE Transactions on Robotics) [<https://journals.ieeeauthorcenter.ieee.org/create-your-ieee-journal-article/authoring-tools-and-templates/ieee-article-templates/templates-for-transactions>]
* Report should not exceed 4 pages, ***not*** counting references. Recommended organization: 1) abstract and introduction: ***0.75*** page; 2) Technical approach: ***1*** page; 3) Results, analyses and discussions: ***2*** pages; 4) Conclusions and discussions: ***0.25*** page.
* Project evaluation will be done on ***a comparative basis to encourage exploration***: *meaning that you will be compared with others choosing the same project topic*.
* Each team must select a topic from the 5 listed below.
* Each team must also submit a “***team effectiveness report***” in pdf format, with a description ***and*** percentage of contribution from each team member (please include names and IDs). ***It must also include a version number of your submission in the title, such as “team effectiveness report (V1.0) ”.*** Each teammate must ***sign*** the team effectiveness report ***with your name AND the version number***. Electronic signatures are fine. This report must be included in your submission package, along with code and a README file.
* If your team submit multiple times (versions), the team effectiveness report must be recreated. ***The submission with the latest version (and by the deadline) that satisfies the above requirements will be considered.***

**Project topic 1. Bi-directional search**

1. Implement bi-directional search describe in the following paper: “*Bidirectional Search That Is Guaranteed to Meet in the Middle*”, Robert C. Holte, Ariel Felner, Guni Sharon, Nathan R. Sturtevant, AAAI 2016, and ***integrate*** into the Pacman domain for path-finding problems (from start to a fixed goal location) in your individual project 1 (<http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/download/12320/12109>)
2. Compare the algorithm’s performance with other search methods (BFS, DFS and A\*) in terms of their search behaviors (e.g., which nodes are expanded first and last). Also, compare them in environments *of* ***different sizes and complexities***. Provide ***statistical analyses*** for your comparison results with different sizes and complexities.
3. Submit a written report with your findings, which should include *at least the following*: 0. Abstract; 1. Introduction (motivation and your achievements in this project); 2) Technical approach (a brief technical discussion of the bi-direction search method you implemented); 3) Results (***Results that CLEARLY illustrate the strengths of your approach compared to others in a statistically meaningful way***, for example, you could compare the number of nodes expanded, computational time, etc.). 4) Conclusions (any observations and discussions.
4. Submit your code with ***comments, and with*** ***instructions*** (as a README file) to run it.

**Project topic 2. Life-long planning**

1. Implement lifelong A\* search described in the following paper: “*D\* Lite*”, Sven Koenig and Maxim Likhachev, AAAI 2002, and ***integrate*** into the Pacman domain for path-finding problems (from start to a fixed goal location) in your individual project 1 (<http://www.aaai.org/Papers/AAAI/2002/AAAI02-072.pdf>). 1) Assume that Pacman only knows about the size of the (grid-world) environment initially, and can ***only observe local environment*** surrounding itself (you may define what local environment is). 2) Also, assume that Pacman always ***knows where it is*** in the environment (i.e., it can localize). 3) Once the Pacman observes something, it is able to ***keep it in its mind*** (i.e., it maintains the knowledge that there is an obstacle in a given location once that is observed).
2. Compare the algorithm’s performance with other search methods (BFS, DFS and A\*) in terms of their search behaviors (e.g., how many times are each node expanded). Also, compare the algorithm’s performance with *an A\* baseline that simply replans every time when new obstacles are observed* in environments *of* ***different sizes and complexities***. Provide ***statistical analyses*** for your results.
3. Submit a written report with your findings, which should include: 0. Abstract; 1. Introduction (motivation and your achievements in this project); 2) Technical approach (a brief technical discussion of the life-long planning method you implemented); 3) Results (***Results that CLEARLY illustrate the strengths of your approach compared to others in a statistically meaningful way***, for example, you could compare the number of nodes expanded, computational time, etc.). 4) Conclusions (any observations and discussions).
4. Submit your code with ***comments, and with*** ***instructions*** (as a README file) to run it.

**Project topic 3: Games**

1. Implement Monte-Carlo tree search as described in this paper:

<https://www.aaai.org/Papers/AIIDE/2008/AIIDE08-036.pdf>

for our Pacman Project 2

1. Evaluate your Pacman agent using Monte-Carlo tree search as more simulations (more node expansions in the MC tree) are played.
2. Compare the performance of Monte-Carlo tree search with our project 2 solution that uses evaluation function. You must provide ***statistical analyses*** for your comparisons.
3. Submit a written report with your findings, which should include: 0. Abstract; 1. Introduction (motivation and your achievements in this project); 2) Technical approach that discusses details of your approach (e.g., how you choose exploration and exploitation in MC tree search; how you expand the tree; any heuristics you used); 3) Results (***Results that CLEARLY support your conclusions in a statistically meaningful way***.). 4) Conclusions (any observations and discussions).
4. Submit your code with ***comments, and with*** ***instructions*** (as a README file) to run it.

**Project topic 4: Logic agent for Minesweeper**

1. Write a logic agent for the Minesweeper domain that assists in game playing by suggesting the next moves. Using a similar representation as what we saw in Project 3 (e.g., drawing the environment in a text-based format). You must ***use the code base and reasoning framework (i.e., minisat) that we used in Project 3*.**
2. Similar to Project 3, provide different playing options for the user: 1) manually; 2) assistive (when a logic agent recommends actions for the next step).
3. Compare your method with a method that randomly selects the next actions ***and*** a method that use some heuristic action selection scheme (that you must design).
4. Submit a written report with your findings, which should include: 0. Abstract; 1. Introduction (motivation and your achievements in this project); 2) Technical approach (a brief technical discussion of the domain, KB rules that you used); 3) Results (***Results that CLEARLY support your conclusions in a statistically meaningful way***.). 4) Conclusions (any observations and discussions).
5. Submit your code with ***comments, and with*** ***instructions*** (as a README file) to run it.

**Project topic 5. Reinforcement Learning agent**

1. ***Implement True Online Sarsa(λ) (eligibility traces, Chapter 12, draft version seems to be free) from Reinforcement Learning: An Introduction, 2nd by Richard Sutton*** method to control your agent in the Pacman domain. This project is based on our project 4 (to be released) on Reinforcement learning.
2. To complete this project, the ***prerequisite*** is that you ***know the basics of reinforcement learning*** [Textbook Chapter 21] and have finished project 4, which we will release soon. ***To compensate for this, this project is comparatively easier than the other projects.***
3. ***Compare the performance of your agent with the reinforcement reasoning agent (with linear function approximation) in project 4. You must compare how fast they converge.*** You must provide ***statistical analyses*** for your results. You must also provide ***an analysis of convergence*** for your learning agent (i.e., how fast it converges to the optimal policy).
4. Submit a written report with your findings, which should include: 0. Abstract; 1. Introduction (motivation and your achievements in this project); 2) Technical approach (a brief technical discussion of ***Sarsa(λ)***); 3) Results (***Results that CLEARLY support your conclusions in a statistically meaningful way***.). 4) Conclusions (any observations and discussions).
5. Submit your code with ***comments, and with*** ***instructions*** (as a README file) to run it.