Problem Set 2: Cooperative AI and the Future of Mechanism Design

Please write an essay to reflect on the following questions in Markdown stored in a ".md" file in the private GitHub repository that I shared privately with you in the CSEcon GitHub organization: https://github.com/CSEcon. Please check duplications and revise iteratively. (Sakai-Week 5-Problem Set 2 Turnitin)

I.Reflection questions and references.

1.What is cooperative AI? What do you see as the potential that computer science and economics can jointly contribute to advance cooperative AI?

Please refer to Prof. Vincent Conitzer's guest lecture and the following readings.

- **Cooperative AI Institution:**
  - Seminar: https://www.cooperativeai.com/seminars
  - YouTube: https://youtu.be/lhOCTVYyHB4
  - Twitter: https://twitter.com/coop_ai
- **Focal@CMU (Prof. Vincent Conitzer's new lab)**:
  - Website: https://www.cs.cmu.edu/~focal/
  - Publication: https://www.cs.cmu.edu/~focal/publications.html
- **NeurIPS 2021 workshop:**
  - https://blog.neurips.cc/2021/08/20/neurips-2021-workshop-announcement/
- **Open Problems in Cooperative AI:**
  - https://arxiv.org/abs/2012.08630
- **Nature Commentary:**
  - https://www.nature.com/articles/d41586-021-01170-0

2. Besides the desirable outcomes of cooperation, what other desirable outcomes that mechanism design theory aims at achieving?

Please refer to Prof. Vincent Conitzer's guest lecture and the following readings.

- Textbook I Chapter 9 and 10 [URL] Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations {Download Free Electronic Version; Cambridge University Press Website}
- Nobel Prize 2007 Leonid Hurwicz, Eric S. Maskin and Roger B. Myerson "for having laid the foundations of mechanism design theory" [URL]

3. What are the limitations of the current mechanism design theory in achieving desirable outcomes? *Please refer to Prof. Vincent Conitzer's guest lecture and the following readings.*

- Arrow's Impossibility[URL]
- The Gibbard-Satterthwaite Impossibility Theorem [URL]

4. **What new challenging are we facing in a new era with more and more human and AI interactions?**

- Open Problems in Cooperative AI: https://arxiv.org/abs/2012.08630

5. Based on your interest and your advantage in skills, how do you plan to contribute to overcoming the limitations of the current mechanism design theory in achieving desirable outcomes and making this world a better place?

I.Format requirements:

★ **Markdown Formatting**

1. For all sources, please provide both in-text citations with hyperlink and endnote bibliographies in chicago author-date style.
2. Must provide a three column glossary table as an appendix (before the end-note bibliography)
3. Utilize headings for structure.
4. Utilize ordered and unordered list to demonstrate points.
5. Apply different font styles to stress key words and jargons.
6. Must include one feature illustration (and you are encouraged to provide more illustrations for each parts: one figure means one thousand words.)
7. Must include a footnote.
8. You must experiment with at least one more markdown formatting besides 1-6 such as adding highlight for a sentence or utilizing blockquote.

★ **Document title information:**

Problem Set 2: Cooperative AI and the Future of Mechanism Design

Author: Your First Name, Last Name
Class, Major, and Affiliation: Your Class, Your Major, Duke Kunshan Univesity

Disclaimer: Submissions to Problem Set 2 for COMPSCI/ECON 206 Computational Microeconomics, 2022 Spring Term (Seven Week - Second) instructed by Prof. Luyao Zhang at Duke Kunshan University.

II. Scaffolding resources:

A code notebook, in general, includes two parts: text and code cells. With the lucid communication of our notebook to others, we provide professional markdown for the text cells and code formatting for the code cells.

The first principle is to make your notebook coherent in logic, self-content in glossaries, and comprehensive in references. A useful strategy is to "put yourself in the others' shoes." Imagine how you would understand your own notebook without any prior information.

For *markdown*, we can refer to the guide (https://www.markdownguide.org/) for some basic golden rules:

The definition: John Gruber (2004) creates Markdown as a lightweight markup language to add formatting elements to plaintext text documents. Markdown is widely used, for example on Reddit and GitHub. Dilinger is one of the best online Markdown editors, that convert the markdown written down in a .md file into HTML. Using markdown, you can typeset basic syntax such as:

- [ ] Headings for structure: # Heading one; ## Heading two; ### Heading three
- [ ] Font style: **text** for bold; *text* for italics
- [ ] Hyperlink: [text](URL)
- [ ] Horizontal Rule: ---
- [ ] Blockquote: >
- [ ] Ordered List and Unordered list
- [ ] Code: `content`
- [ ] Image ![alt text](image. jpg)

You can refer to the following documents for more typesetting options:

- [ ] Cheat-sheet: https://www.markdownguide.org/cheat-sheet/
- [ ] Basic Syntax: https://www.markdownguide.org/basic-syntax/
- [ ] Extended Syntax: https://www.markdownguide.org/extended-syntax/
- [ ] Hacks: https://www.markdownguide.org/hacks/

Here are the additional references for creating markdown:

- [ ] John Gruber's Markdown documentation. The original guide was written by the creator of Markdown.
- [ ] Markdown Tutorial. An open-source website that allows you to try Markdown in your web browser.
- [ ] Awesome Markdown. A list of Markdown tools and learning resources.
- [ ] Typesetting Markdown. A multi-part series that describes an ecosystem for typesetting Markdown documents using pandoc and ConTeXt.
- [ ] A Crash Course: https://www.youtube.com/watch?v=HUBNt18RFbo

- [ ] The Github: https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax

Here is the markdown authoring applications:

- [ ] **Mac:** MacDown, iA Writer, or Marked 2
- [ ] **iOS / Android:** iA Writer
- [ ] **Windows:** ghostwriter or Markdown Monster
- [ ] **Linux:** ReText or ghostwriter
- [ ] **Web:** Dillinger or StackEdit

For *code formatting*, the golden rules are to improve expandability. You can confirm the checklist:

- [ ] Write your code in a simple and logical structure (e.g., following the PEP-8 Standard)
- [ ] Write *comments* beginning with a hash (number sign) (#) for signal line explanations
- [ ] Write *docstring* that describes modules, classes, and functions

"""

docstring

docstring

docstring

"""

- [ ] Add necessary indention, black spaces, line spaces manually or using the tools such as "black"

- [ ] you can refer to
  - [ ] **PEP-8 Standard**: https://realpython.com/python-pep8/
  - [ ] **Format the code with black**:(Blanks) https://github.com/psf/black (YouTube)
  - [ ] **Github documentation**: https://github.com/realpython/python-guide/blob/master/docs/writing/documentation.rst#id19

Finally, you can also consider Sphinx (https://www.sphinx-doc.org/en/master/) for intelligent and beautiful (python) documentation.