



COMPUTATIONAL APPLICATIONS TO POLICY AND STRATEGY (CAPS)

Session 3 – Building a Rule-Based StarCraft II Bot

Leo Klenner

Outline

1. Recap
2. StarCraft II Gameplay
3. Strategy for CapsBot
4. Debrief
5. Required Software



Recap – Sarah Sewall on AI Ethics

- > Five year window for U.S. to build and implement holistic AI framework
- > Intersection of AI and IR demands nuanced and complex action
- > Need for collaborative platform and open debate on AI ethics and strategy
- > Strong demand for AI-international-x
- > Translating between engineers and policymakers considered valuable skill



Recap – Decomposing WorkerRushBot

```
import sc2
from sc2 import run_game, maps, Race, Difficulty
from sc2.player import Bot, Computer
```

} load source components

```
class WorkerRushBot(sc2.BotAI):
    async def on_step(self, iteration):
        if iteration == 0:
            for worker in self.workers:
                await self.do(worker.attack(self.enemy_start_locations[0]))
```

} create bot and specify operations

```
run_game(maps.get("Abyssal Reef LE"), [
    Bot(Race.Terran, WorkerRushBot()),
    Computer(Race.Protoss, Difficulty.Medium)
], realtime=True)
```

} initialize game environment and execute bot

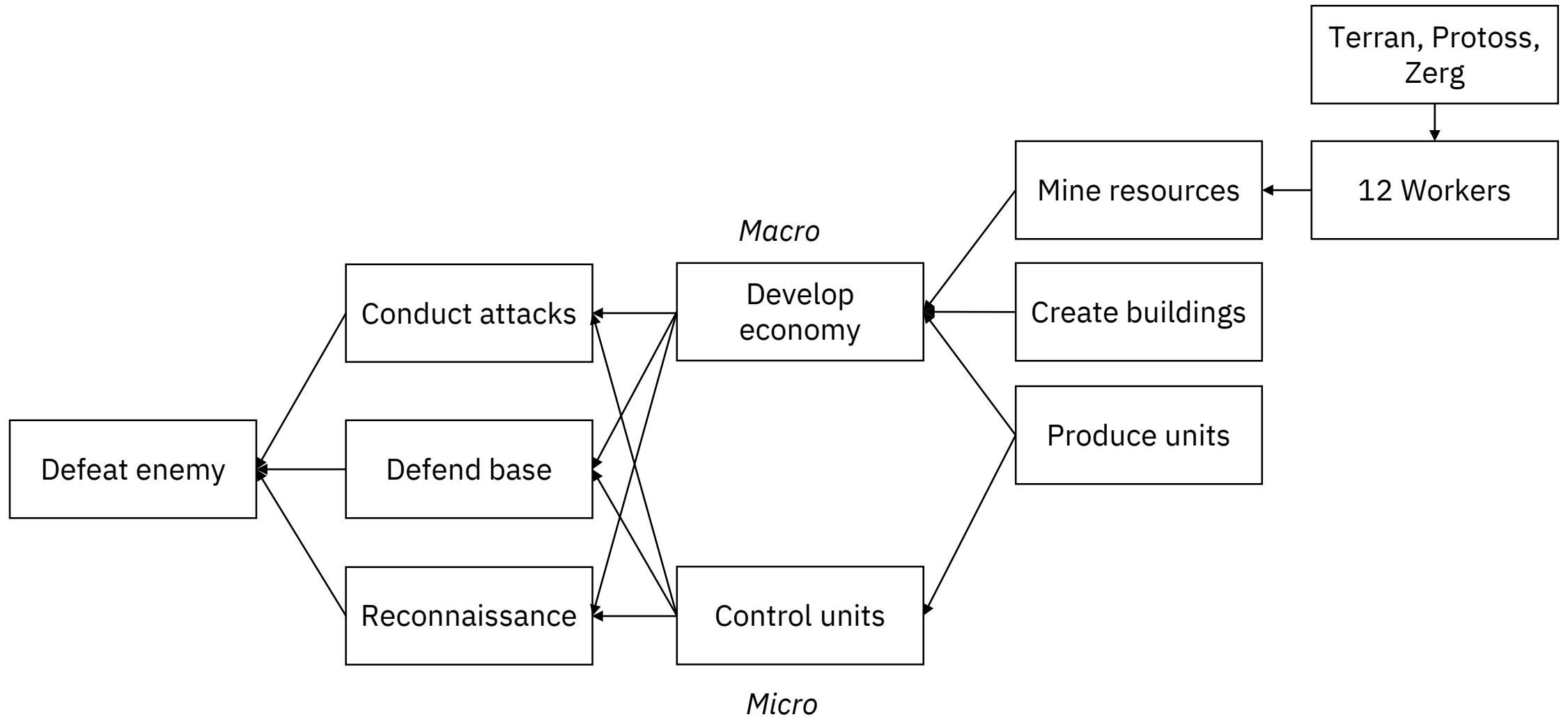


Goals

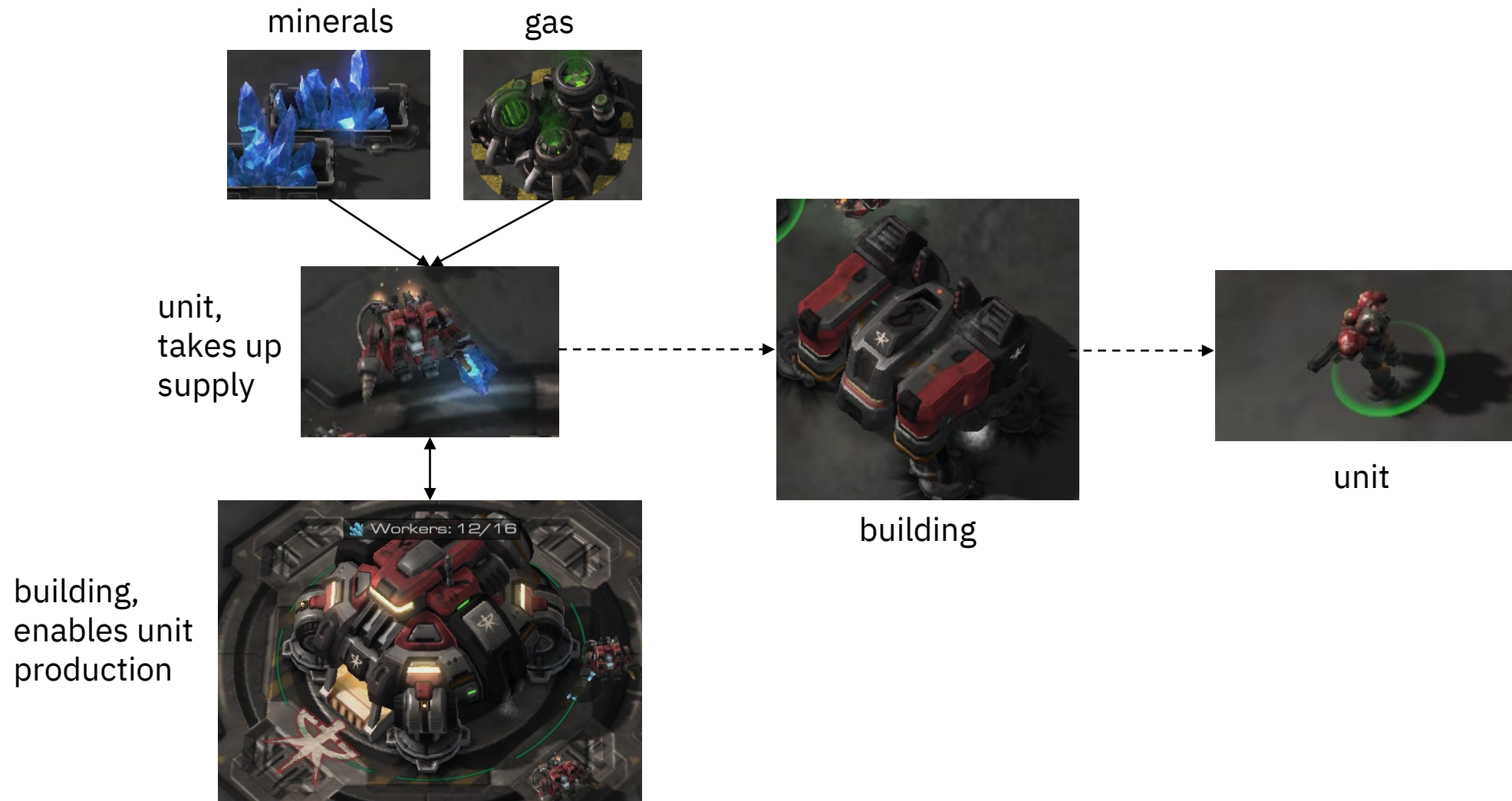
- > Write CapsBot ~100 lines of code to defeat hard built-in AI
- > Conceptualize the performance of CapsBot
 - > Decision-making under uncertainty
 - > Predictability
 - > Operational constraints



StarCraft II Gameplay – Overview



StarCraft II Gameplay – Chain of Production



Gameplay – Strategy

- > Strategy shaped by complex dependency trees
- > Adaptation based on scouting
- > Early game, mid game, late game
- > Micro v. macro-intensiv play

- > General pattern
 - > Resource production and scouting
 - > Fights over map control
 - > Max supply, all-in battles between upgraded armies



Gameplay Example

> Maru v sOs (replay video)



Strategy for CapsBot

- > Marine and Cyclone rush at around 5:30 min into the game
 - > Overwhelm enemy in one push with 20+ Marines and 4+ Cyclones
 - > Reinforce continuously until minimum unit level is reached
 - > If minimum reached, pause reinforcement to build up second push



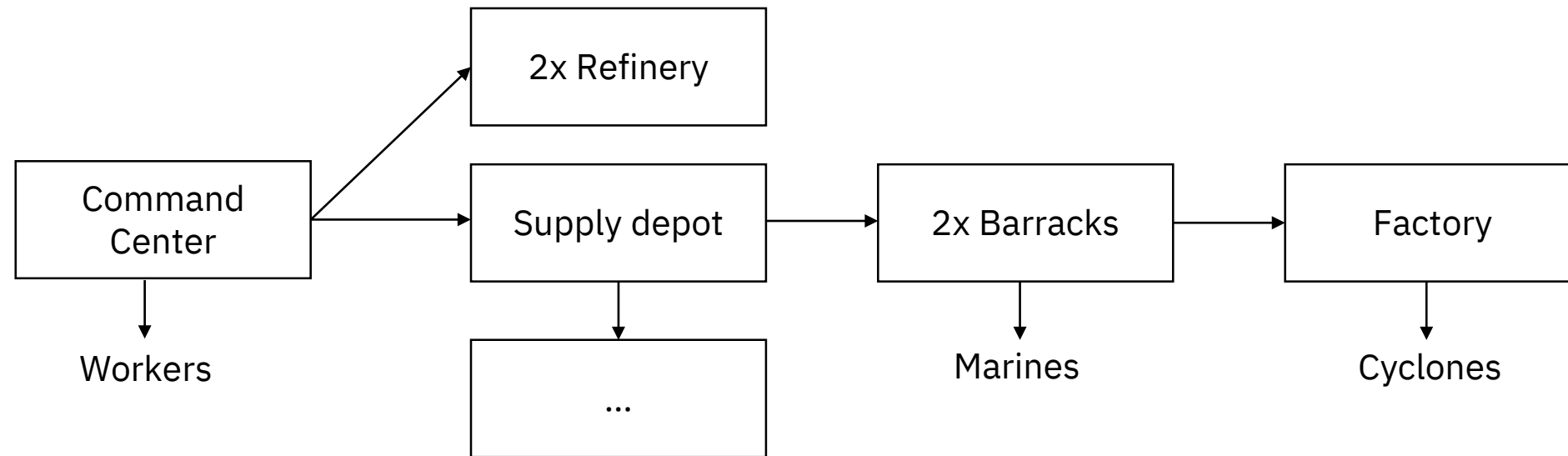
Marine



Cyclone

Fast, expendable and versatile

Build Order for CapsBot



Code for CapsBot

> <https://github.com/SAIS-S2S-Technology/Roadmap/blob/master/CAPS/CapsBot.py>

> <https://github.com/SAIS-S2S-Technology/Roadmap/blob/master/CAPS/CapsBot.py>



Debrief

- > Performance
- > Decision-making
- > Robustness
- > Predictability



Constraints



- > Randomized building placement can lead to substantial trade-offs

Applications of Simple Logic



- > Building new supply depot next to last one creates protective wall

Summary

- > We built a rule-based agents through a set of iterations to defeat the built-in AI of StarCraft II on hard difficulty
- > The performance of the rule-based agent is succesful in the environments we provided, but not necessarily robust
- > Although rule-based agents always perform the same actions, changes in the environments will transform these actions



Required Software

- > For next week you only need pySC2 (DeepMind)
- > From command line
 - > `pip install pysc2`

