CMSC 722, AI Planning
Programming Project:
Comparing Domain-Independent and HTN Planners

Intermediate progress report due March 10 (late date March 12)

Final report due April 7 (late date April 9)

Overview

- Objective
  - Compare a domain-independent planner with an HTN or HGN planner
- I'll provide links to
  - Several domain-independent planners and HTN/HGN planners
  - PDDL definitions and problem generators for two planning domains
- You should
  - Choose a domain-independent planner and an HTN or HGN planner
  - Write code to get the domains working properly
  - Compare the two planners experimentally on the two domains
  - Submit a progress report about half-way through the project
  - Write a report about your work and the results, to submit at the end of the project

Planners

- For the HTN or HGN planner, use either Pyhop or GTPyhop

- Use one of the domain-independent planners in this list. Here are some promising ones:
  - Fast Downward (and various modified versions of it)
  - MetricFF
  - LPRPG

Planning Domains

1. The blocks world

- A PDDL definition
- A web page for generating random blocks-world problems and solving them
  - I'll put C source code for the problem generator on Piazza

- Your domain-independent planner will need PDDL, but the problem generator doesn't produce it
  - You'll need to write a translator to translate the problem generator's output into PDDL
    - OK to collaborate with others on this, but **only** this
    - If you do so, your report should say who did what

- Pyhop includes an HTN blocks-world domain representation (methods, actions, example problems)
- GTPyhop includes three blocks-world domain representations: HTN, HGN and hybrid HTN/HGN
- You'll need to do the following:
  - Decide which planner and domain representation to use
  - Write code to translate the problem generator's output to something compatible with the domain representation you chose

Planning Domains

2. The Satellite Domain (metric version)

- A PDDL definition
- On Piazza, I've put C source code for a problem generator that generates random problems
  - It produces PDDL output that I think is compatible with the above problem definition
  - You should check to make sure

- For Pyhop or GTPyhop, you need to do the following:
  - Decide on a domain representation
    - States, actions, tasks or goals
    - Write HTN or HGN methods
  - Write code to translate the problem generator's output to something compatible with your domain representation

Experimental Comparisons

- Objective: measure performance as a function of problem size
  - For the blocks world, *size* = number of blocks
  - For the Satellite domain, *size* = number of targets
    - Other parameters: number of satellites (10?), max instruments per satellite (10?), number of modes (5?), number of targets *(size),* number of observations (2?)
    - With the above parameters, the command line will be

      `satgen -n 10 10 5` *size* `2`

- Generate test suites to compare the planners on
  - Randomly generated problems
  - At least 10 different problem sizes, and at least 10 problems of each size
    - More if feasible
- For published papers I often do several hundred problems of each size, but I won't ask you to do that

Performance Metrics

1. Length of plan

- Easy to measure

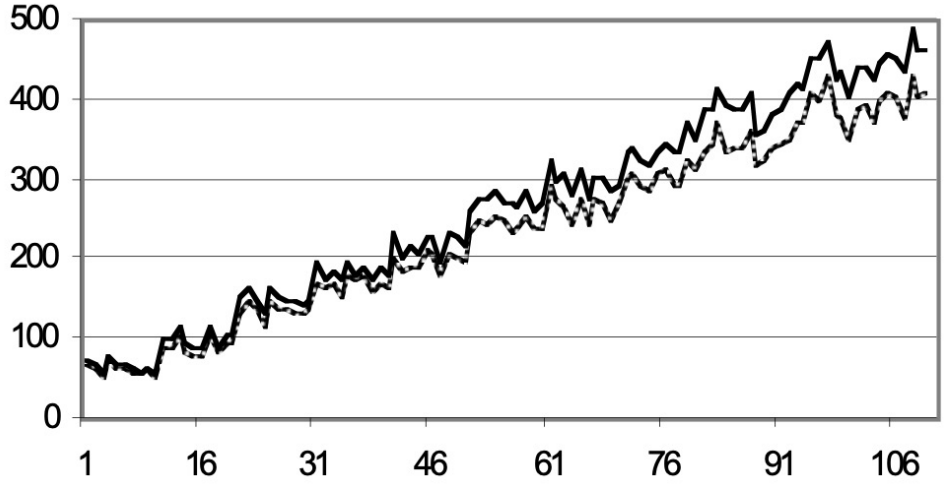
2. How much work the planner did to find the plan

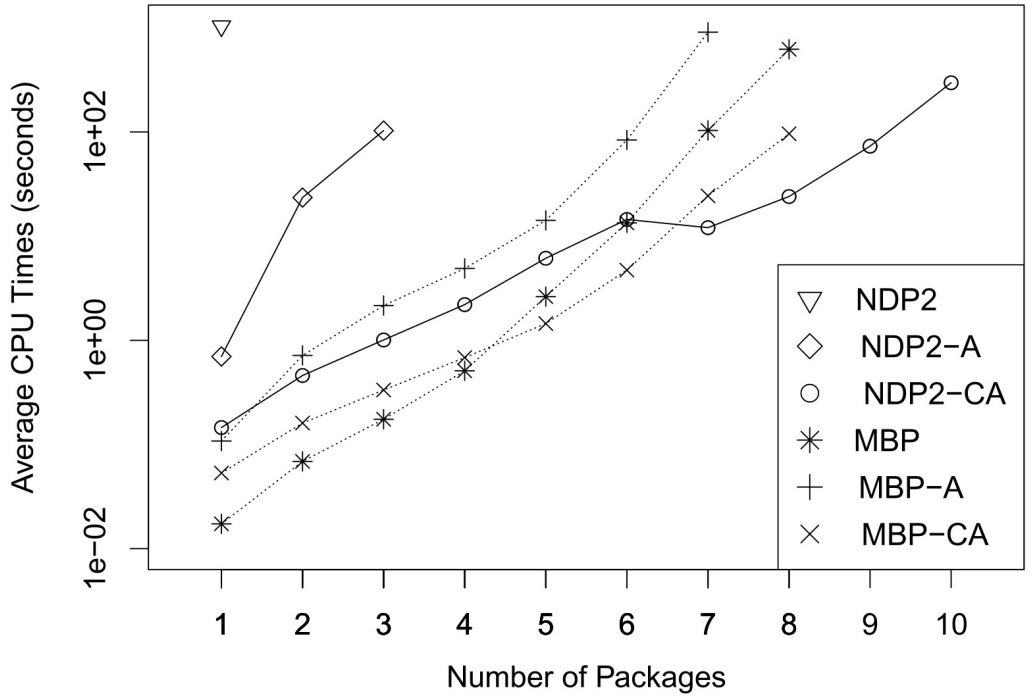
- CPU time? Number of nodes expanded? Neither is a fair comparison
  - CPU time favors highly optimized planners (not Python-based)
  - Number of nodes favors HTN/HGN rather than domain-independent
- Measure them anyway, but try to measure rate-of-growth over a large number of problem sizes
  - As many as you think you need to get good results

- In an HTN/HGN planner, what things are nodes? States? Tasks? Methods? Actions?
  - Figure out a reasonable answer, justify it in your report

Plotting your data

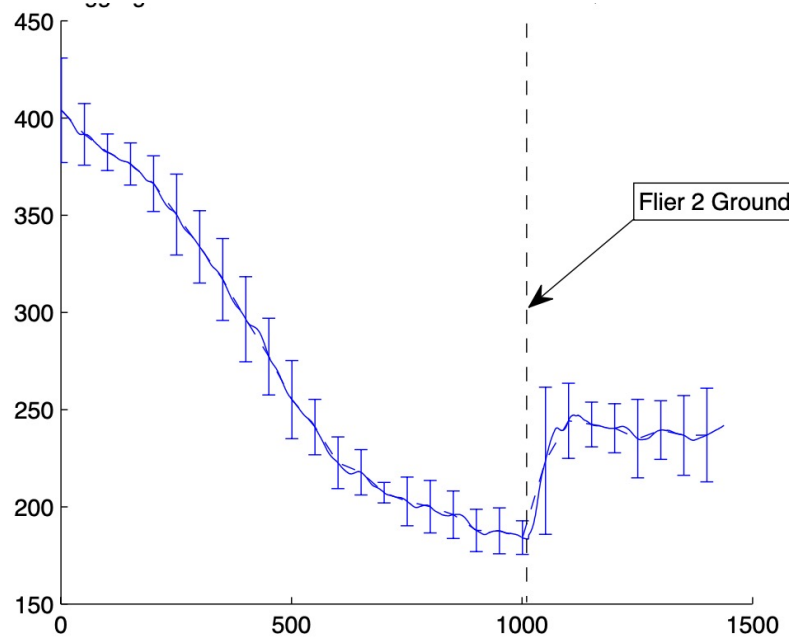
- Line plot, one line for each planner
  - x axis = problem size
  - y axis = performance
    (CPU time, nodes expanded, plan length)
- Good for showing two things:
  - relative performance, and rate of growth
- If rate of growth is exponential, use semi-log plot
- Drawback: some problems of size *n* may be much harder to solve than others
  - This may throw some of the measurements off

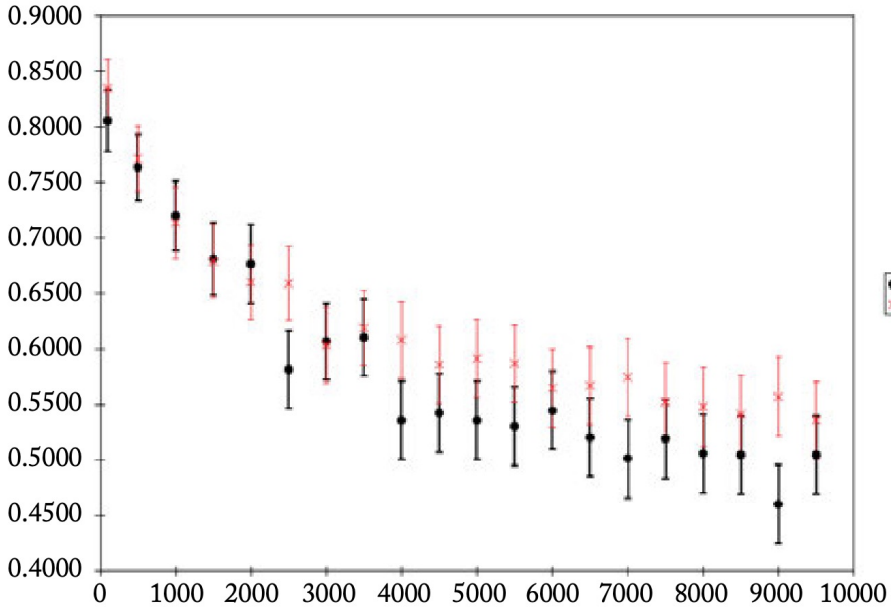


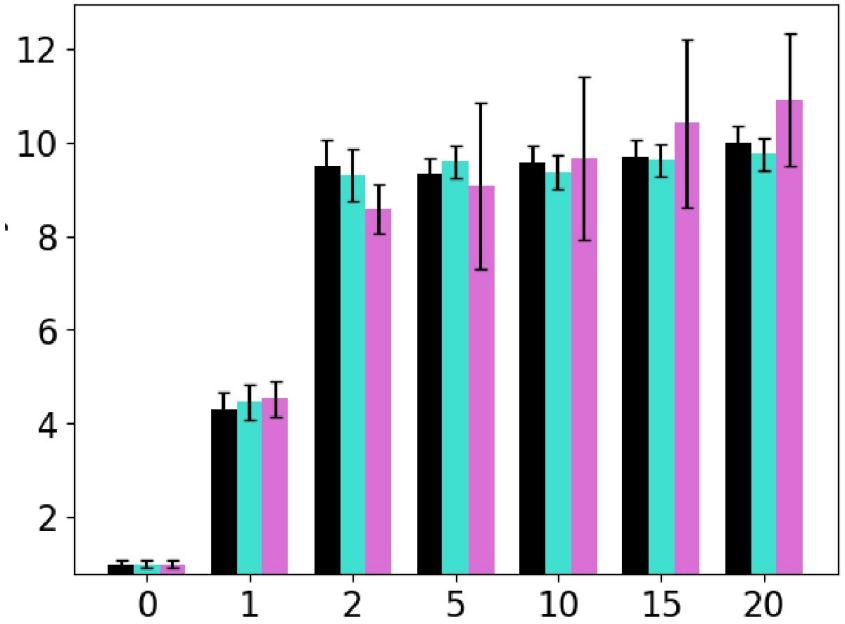


Plotting your data

- Drawback with line plots
  - Some problems of size *i* may be much harder than others
  - This may throw some of the measurements off
- Good idea to show error bars (confidence intervals)
  - Problem: error bars for the two planners may overlap
  - If so, a bar chart may be better

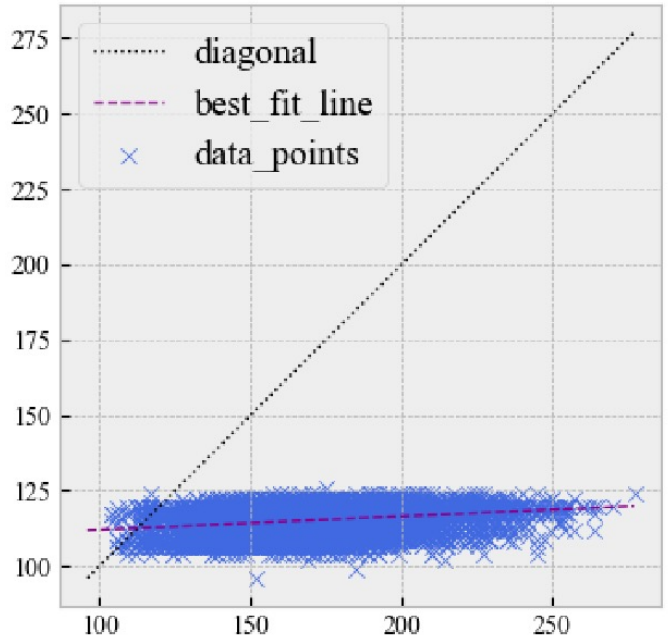






Plotting your data

- Scatter plot
  - x axis = performance measurement for one planner
  - y axis = performance measurement for the other planner
- Advantage
  - Doesn't matter if some problems of size *i* are harder than others
- Disadvantage:
  - Doesn't show rate of growth, just relative performance



- Suggestion:
  - Try plotting your results several different ways
  - Choose whichever way(s) are most effective

Final Report

- Write a final report giving the results of your experiments.
  - Format: US letter paper, one column, 1-inch margins, font size at least 11pt
- Include the following items:
  - Title page
    - Abstract
    - Signed copy of the student honor code
  - Introduction:
    - Summarize what you did
  - Description of the planners:
    - Tell what domain-independent planner you chose, and why
    - Describe how your HTN or HGN methods work
  - Experiments:
    - Describe experimental design
    - Include plots showing your results
    - For each plot, tell what you can conclude from it and why
  - Conclusions:
    - overall meaning and significance of the results
  - Supplemental data: ZIP file containing
    - all of the programs and data needed to duplicate your results
    - instructions on how to run them

Intermediate Progress Report

- You have six weeks to complete the project
- After the first three weeks, you should have accomplished the following tasks:
  - Choose a domain-independent planner
  - Get it to run correctly on both planning domains
    - For each planning domain, run it on at least one problem of each size

- Write an informal report describing your progress
  - Tell what domain-independent planner you chose, and why
  - For each domain, include one plot
    - x-axis = problem size, y-axis = CPU time
    - Run the planner on one problem of each size
    - Since it's just one problem of each size, don't include error bars
  - If you've made additional progress beyond what I listed above, briefly describe it

Grading Criteria

- Evaluation criteria
  - Progress report: 25%
    - Demonstration of progress
    - Clarity of presentation
  - Your final report: 60%
    - Your HTN/HGN methods and actions
    - Your experiments
    - Clarity of presentation
    - Quality of conclusions
  - Supplemental material: 15%
    - Code and documentation
    - Raw data - experiments and results
- Extra credit if you do something especially impressive
  - e.g., especially good HTN/HGN methods for the Satellite domain