Personal Background

- Image processing with pyramid data structures and parallel architectures (1973–1989)
- "The Elements of Artificial Intelligence Using Lisp" textbook (1987, 90, 95)
- "Mathematics Experiences Through Image Processing" (1992–present)
- Visual languages (1983–present)
- Collaborative problem solving (2009–present)

Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues
Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues

Motivation

- The need to solve problems never ends.

- “All life is a series of problems which we must try and solve. First one, and then the next, and the next, until at last we die.” -- Old Lady Grantham in Downton Abbey.
“Life is a series of problems that we must try and solve...”
—Old Lady Grantham, in Downton Abbey

Problems are Getting Harder

- Global warming
- Weapons proliferation
- Population growth: food, sustainability
- Information glut and info. pollution
- Drug-resistant diseases
Problems are Getting Harder

“Wicked Problems” (Rittel & Webber, 1973)

- Global warming
- Weapon proliferation
- Population growth: food, sustainability
- Our wealth of information is getting more complex
- Diseases are resisting treatment
- Social patterns are evolving more quickly

Cross-Cutting Research

- Design of systems for collaborative problem solving is CROSS-CUTTING
- Multidisciplinary
- Interdisciplinary
- Integrative
- HCI, AI, Design, Socio-Computational Systems, Software Architecture, Mathematics
US National Science Foundation’s Interdisciplinarity

“NSF has long recognized the value of interdisciplinary research in pushing fields forward and accelerating scientific discovery. Important research ideas often transcend the scope of a single discipline or program.”

“NSF also understands that the integration of research and education through interdisciplinary training prepares a workforce that undertakes scientific challenges in innovative ways.”

Outline

• The Need to Solve Problems
• Polya and Metaproblem solving
• Existing Systems that Support Problem Solving
• Classical Theory of Problem Solving
• TStar Examples
• The CoSolve System
• Overview of Issues
Inspiration from George Polya

*How To Solve It* (1945) -- Deliberate Methodologies for Solving Problems

Motivation (cont.)

As computer technology improves, it’s important to ask (again and again) what computing can do to help solve problems.
Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues

Generic Collaboration Tools

- Google Drive (Google Docs)
- Email
- Forums and Blogs (Tim Gowers used a blog for the Polymath project)
- Skype, MS Remote Desktop, etc.
Existing Systems for Problem Solving

INOCENTIVE

GALAXY ZOO

koios

amazon mechanical turk

foldit

Collaborative Problem Solving

Existing Systems

INOCENTIVE

GALAXY ZOO

koios

amazon mechanical turk

foldit

Missing in each: 1 or more of Collaboration, Computation, Generality

Collaborative Problem Solving
• Harnessing the power of the crowd for industrial problem solving
• Business model: multiple invention competitions.
• Value added by Innocentive: Matchmaking.

“Polymath” Project (Tim Gowers)

Gowers's Weblog

• 2009 – mathematics blog post.
• Invited the public to help find a new combinatorial proof to the density version of the Hales–Jewett theorem.
• Also asked, “is massively collaborative mathematics possible?”
Multidimensional Tic-Tac-Toe

What Issues Arise In Designing General CSCPSS?

- Support PS in a deep way
- Support collaboration
Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues

The Classical Theory of Problem Solving

Began with the General Problem Solver (GPS) G.W. Ernst, Alan Newell & Herbert Simon. The CTPS is well described in several AI textbooks, e.g.,

Definition

- A problem is a triple: \((\sigma_0, \Phi, \Gamma)\) where \(\sigma_0\) is an initial state, \(\Phi\) is a set of operators, and \(\Gamma\) is a set of goal states.
- Each \(\phi_i \in \Phi\) has a precondition, a state-transformation function, and an optional parameter list.
- These implicitly define \(\Sigma\), the set of all states reachable from \(\sigma_0\) by applying members \(\phi_i \in \Phi\) zero or more times.
Example: Towers of Hanoi

- \( P = (\sigma_0, \Phi, \Gamma) \)
- \( \sigma_0: \)

- \( \Phi: \{ \text{Move1}_2, \text{Move1}_3, \text{Move2}_3, \text{Move2}_1, \text{Move3}_1, \text{Move3}_2 \} \)
- \( \Gamma = \{ \gamma \}: \)

Why do I use puzzles?

Judea Pearl:

“The expository power of puzzles and games stems from their combined richness and simplicity. If we were to use examples taken from real-life problems, it would take more than a few pages just to lay the background...”
Problem Solving = State–Space Search

Advantages of the Classical Theory

- understandable to non-technical people.
- provides a solid, well-understood foundation for implementing systems.
- supports higher-level representations: constraints, probability distributions, implicit state-spaces, etc.
- allows autonomous agents.
Outline

• The Need to Solve Problems
• Polya and Metaproblem solving
• Existing Systems that Support Problem Solving
• Classical Theory of Problem Solving
• TStar Examples
• The CoSolve System
• Overview of Issues

Precursor to CoSolve: TStar

Transparent STate–space search ARchitecture:

A Python toolkit for exploring human-guided state space search.
TStar Example: Missionaries and Cannibals

TStar Example: Musical Motif Creation
Limitations of TStar

- Not collaborative in a direct way.
- Not web-based.
- Limited affordances for using agents.

Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues
CoSolve: A web-based system for collaborative problem solving.

- Emulates TStar:
  - Displays a tree of the explored portion of the state space.
  - Permits user(s) to apply operators and generate new states.
  - Scripted using Python.
  - Problem-solving sessions are saved.

CoSolve (cont)

- Surpasses TStar:
  - Supports a GROUP of users within a session.
  - Keeps track of authorship of states.
  - Offers multiple viewers: HTML (browser) and Flash-based.
  - All sessions are persistent by default.
  - Server-based.
  - Supports “posing” of new problems.
Video Demonstrating CoSolve

- Shows the “solving” process (not posing).
Work with Users on Solving
(conducted by Sandra Fan and Tyler Robison)

How do people use a CoSolve-like system for collaborative problem solving?

The CitySim Problem Template

- Balances simplicity, novelty, and richness.
- Design a city to maximize tax income.
- Place buildings that have different effects on your citizens.
User Study Procedure

- Background questionnaire
- 20–30 minute tutorial of system and CitySim
- 1.5 hour solving session
- Individual post–activity interviews and wrap–up questionnaire

Overall Team Performance

**TABLE 1. TEAM RESULTS**

<table>
<thead>
<tr>
<th>CitySim Score</th>
<th>Team 1</th>
<th>Team 2</th>
<th>Team 3</th>
<th>Team 4</th>
<th>Team 5</th>
<th>Team 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>33087</td>
<td>31314</td>
<td>26576</td>
<td>14848</td>
<td>26865</td>
<td>20352</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of Nodes in Tree</th>
<th>178</th>
<th>264</th>
<th>399</th>
<th>392</th>
<th>320</th>
<th>101</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of Annotations</td>
<td>38</td>
<td>21</td>
<td>25</td>
<td>22</td>
<td>5</td>
<td>55</td>
</tr>
</tbody>
</table>

- Positive Annotations: 17 | 7 | 18 | 7 | 4 | 5
- Neutral Annotations: 20 | 12 | 7 | 15 | 0 | 48
- Negative Annotations: 1 | 2 | 0 | 0 | 1 | 2
**Team 3’s tree**

**Interview and Behavior Results**

- Tree visualization useful for collaboration because one could learn from other users’ thought processes
- Used thumbs-up annotations to find places to build on others’ work
General Result of Study

Although participants would have liked more navigation affordances, they found the tree visualization of the problem space helpful and not confusing.

Outline

- The Need to Solve Problems
- Polya and Metaproblem solving
- Existing Systems that Support Problem Solving
- Classical Theory of Problem Solving
- TStar Examples
- The CoSolve System
- Overview of Issues
Overview of Issues

- Problem Formulation
- Wicked Problems
- Problem-Space Visualization
- Transparent Interfaces and Design
- Collaboration and Crowdsourcing
- Problem-Solving Agents
- Collaborative Learning
Problem Formulation

- Having a good formulation is a big step towards solving the problem.
- To exploit CoSolve structures, the problem must be organized with clear states, operators, and visualizations (if possible)
- There are many issues in common with software engineering, e.g., iterative design.

Wicked Problems

- Many real-world problems seem to resist formulation, because of ambiguities, inconsistencies, and stakeholder differences.
- Using new tools, many of these difficulties can be handled.
Formulation Approaches

- Scaffolded Programming (CoSolve)
- Manipulable Constraints
- Metaproblem Templates
- Extensible Problems, Virtual Operators, Universal Operators
Formulation as Solving a Metaproblem

Euler’s formula for polyhedra:
For any polyhedron $P$: $V(P) - E(P) + F(P) = 2$

Imre Lakatos: “Proofs and Refutations”

Transparent Interfaces and Design

- Helping users understand complex data structures and systems is a challenge.
- Transparent interfaces can help.
- Collaboration, agents, and complex problem structure all present opportunities for designing effective human–computer interfaces.
Transparency Topics

- Glass-box transparency
- Visualization of AI techniques
- Trust, Self-efficacy, Flow
- Liveness
- Models of Transparency

Problem-Space Visualization

- Visual thinking can support problem solving.
- The design of visualizations for problem spaces is timely.
- The classical theory offers components for automatic or computer-assisted design of visualizations.
Problem Space Mappings

- Distance functions
- Barycentric coordinates
- Heuristics

Collaboration and Crowdsourcing

- How to incentivize solvers?
- How to foster collaboration?
- How to offer specialized roles, well-matched to participant talents and interests?
- How to maintain order within solving sessions involving conflicting goals?
Roles in Problem Solving

- Roles offer structure for collaboration.
- PRIME Game design roles:
  - Architect
  - Image puzzle designer
  - Music puzzle designer
  - Game logic designer
Supporting Collaboration

- Crowds
- Incentives
- Workflows
- Ownership
- Privacy

Problem-Solving Agents

- What can agents do for solvers?
- Who should create and monitor agents?
- How should the work of agents be managed?
- The future of agents
Supporting Computation

- Clouds
- Mental models for agents
- Provisionality of agent work
- More roles for agents
- Scalability questions

Collaborative Learning

- The field of Computer-Supported Collaborative Learning (CSCL) has important potential synergies with collaborative problem solving.
- All solvers of complex problems must be learners.
- Situated problem solving offers great opportunities for learners.
CSCL Challenges

- Assessment of group and individual contributions
- Unobtrusive assessment
- Embedded tutorials, authoring
- Curricula around problem formulation
- Educational vs Real-world problem solving

Jonassen's work on Problem-Based Learning
Recap of the Lecture Schedule

- **Introduction**  }  Feb. 10
- **Problem Formulation**  }  Feb. 17
  + Wicked Problems
- **Transparent Interfaces and Design**  }  Feb. 24
- **Problem-Space Visualization**  }  Mar. 3
- **Collaboration and Crowdsourcing**  }  Mar. 10
  + Problem-Solving Agents
- **Collaborative Learning**  }  Mar. 17

Acknowledgments

- Thank you to hosts.
- Thank you to CoSolve developers and contributors.
- Thank you to the audience.
アリガとうございませ。