Cost-Algebraic Heuristic Search

Stefan Edelkamp*
Shahid Jabbar*
Alberto-Lluch Lafuente**

*LS-5, Computer Science Department
University of Dortmund, Dortmund, Germany.
**Department of Informatics, University of Pisa, Pisa, Italy.
Motivation

To which domain structures can we apply A* search?
Let $A$ be a set and $\times : A \times A \rightarrow A$ be a binary operator. A monoid is a tuple $\langle A, \times, 1 \rangle$ if $1 \in A$ and for all $a, b, c \in A$

(1) $a \times b \in A$  
(2) $a \times (b \times c) = (a \times b) \times c$  
(3) $a \times 1 = 1 \times a = a$  

(closeness) (associativity) (identity)
Let $A$ be a set. A relation $\leq \in A \times A$ is a **total order** whenever for all $a, b, c \in A$

\begin{align*}
(1) & \quad a \leq a \quad \text{(reflexivity)} \\
(2) & \quad a \leq b \land b \leq a \implies a = b \quad \text{(anti-symmetry)} \\
(3) & \quad a \leq b \land b \leq c \implies a \leq c \quad \text{(transitivity)} \\
(4) & \quad a \leq b \lor b \leq a \quad \text{(total)}
\end{align*}
A *cost algebra* is defined as a 6-tuple \( \langle A, \sqcup, \times, \preceq, 0, 1 \rangle \), such that

1. \( \langle A, \times, 1 \rangle \) is a monoid
2. \( \preceq \) is a total order induced by \( \sqcup \)
3. \( 0 = \bigcap A \) (minimum) and \( 1 = \bigcup A \) (maximum)
4. \( A \) is isotone
Isotonicity

A set $A$ is *isotone* if $a \preceq b$ implies both $a \times c \preceq b \times c$ and $c \times a \preceq c \times b$ for all $a, b, c \in A$.

A set $A$ is *strictly isotone* if $a \prec b$ implies both $a \times c \prec b \times c$ and $c \times a \prec c \times b$ for all $a, b, c \in A$, $c \neq 0$. 
Examples

- **Boolean:** \( \{\text{true, false}\}, \lor, \land, \Rightarrow, \text{false, true} \)  
  Network/service availability.

- **Optimization:** \( \mathbb{R}^+ \cup \{+\infty\}, \text{min}, +, \leq, +\infty, 0 \)  
  Price, propagation delay.

- **Max/Min:** \( \mathbb{R}^+ \cup \{+\infty\}, \text{max, min}, \geq, 0, +\infty \)  
  Bandwidth.

- **Probabilistic:** \( [0, 1], \text{min}, \cdot, \geq, 0, 1 \)  
  Performance and rates.

- **Fuzzy:** \( [0, 1], \text{max, min}, \geq, 0, 1 \)  
  Performance and rates.
Definition: The Principle of Optimality requires
\[ \delta(s, v) = \bigsquare \{ \delta(s, u) \times \omega(e) \mid u \xrightarrow{e} v \}, \] where \( s \) is the start node in a given graph \( G \).
Problems with Principle of Optimality

Dijkstra’s algorithm delivers

An optimal path with optimal subpaths

Example: A widest path is a path of maximum capacity, where capacity = \( \min(e) \mid e \in P \)

Both paths \( s-x-t \) and \( s-y-x-t \) are widest paths.

But for \( s-x-t \), \( s-x \) is not a widest path - cost 10.
Reason for this non-optimality

Optimality Principle requires Strict Isotonicity:
\[ a < b \implies a \times c < b \times c \quad \text{and} \quad c \times a < c \times b \quad \text{for all} \quad a, b, c \in A, \quad c \neq 0. \]
Recall Max/Min: \( (\mathbb{R}^+ \cup \{+\infty\}, \max, \min, \geq, 0, +\infty) \)

Counter Example: \( 5 > 3 \), but \( \min\{5, 3\} = \min\{3, 3\} \)

So, to ease our treatment,

let us ease our criteria of optimality
Prefix Optimal Paths

A path $p = u_0 \xrightarrow{e_0} \ldots \xrightarrow{e_{k-1}} u_k$ is \textit{prefix-optimal}, if all of its prefixes, i.e., all paths $u_0 \xrightarrow{e_0} \ldots \xrightarrow{e_{i-1}} u_i$ with $i < k$, form an optimal path from $u_0$ to $u_i$. 
Cost-Algebraic Dijkstra

**Initialization**  
\[ f(u_0) \leftarrow 1; \text{Open} \leftarrow \{u_0\}; \]

for each \( u \neq u_0 \):  
\[ f(u) \leftarrow 0 \]

**Selection**  
Select \( u \in \text{Open} \) with  
\[ f(u) = \bigsqcup \{ f(v) \mid v \in \text{Open} \} \]

**Update**  
\[ f(v) \leftarrow \]

\[ \bigsqcup \{ f(v) \} \cup \{ f(u) \times \omega(e) \mid u \xrightarrow{e} v \} \]

**Proposition:** Cost-algebraic Dijkstra finds prefix-optimal least-cost solution path
Heuristics: Admissible and Consistent

A heuristic function $h : V \rightarrow A$ with $h(t) = 1$ for each goal node $t \in T$ is

- **admissible**, if for all $u \in V$ we have $h(u) \leq \delta(u, T)$, i.e., $h$ is a lower bound.

- **consistent**, if for each $u, v \in V$ and $e \in E$ such that $u \xrightarrow{e} v$, we have $h(u) \leq \omega(e) \times h(v)$.
Cost-Algebraic A*

**Initialization**  
\[ f'(u_0) \leftarrow h(u_0), \]  
\[ \text{Open} \leftarrow \{u_0\}; \text{for each} \]  
\[ u \neq u_0 : f'(u) \leftarrow 0 \]

**Selection**  
Select \( u \in \text{Open} \) with  
\[ f'(u) = \bigsqcup \{ f'(v) | v \in \text{Open} \} \]

**Update**  
\[ f(v) \leftarrow \]  
\[ \bigsqcup \{ \{ f(v) \} \cup \{ f(u) \times \omega(e) | u \xrightarrow{e} v \} \} \]  
\[ f'(v) \leftarrow f(v) \times h(v) \]

**Proposition:** Cost-algebraic A* finds prefix-optimal least-cost solution path
Multi-criteria Search

The Prioritized Cartesian Product of

\( C_1 = \langle A_1, \sqcup_1, \times_1, \preceq_1, 0_1, 1_1 \rangle \) and

\( C_2 = \langle A_2, \sqcup_2, \times_2, \preceq_2, 0_2, 1_2 \rangle, \)

\( C_1 \times_p C_2 \) is a tuple

\[ \langle A_1 \times A_2, \sqcup, \times, \preceq, (0_1, 0_2), (1_1, 1_2) \rangle, \]

where \((a_1, a_2) \times (b_1, b_2) = (a_1 \times b_1, a_2 \times b_2)\),

\((a_1, a_2) \preceq (b_1, b_2) \) iff \( a_1 \prec b_1 \lor (a_1 = b_1 \land a_2 \preceq b_2) \).

Proposition: If \( C_1, C_2 \) are cost algebras and \( C_1 \)
is strictly isotone then \( C_1 \times_p C_2 \) is a cost algebra.
Experiments

- Implemented generalized Dijkstra and A* in C++.
- Made possible by the template facility of C++.
- Random graphs generated by the $G(n, p)$ model with $n$ as the number of nodes and $p$ being the probability of having an edge between two nodes.
- Abstraction heuristic based on node merging.
## Experiments: Optimization

<table>
<thead>
<tr>
<th>nodes</th>
<th>edges</th>
<th>visited\textsubscript{Dij}</th>
<th>time\textsubscript{Dij}</th>
<th>visited\textsubscript{A*}</th>
<th>time\textsubscript{A*}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>30,111</td>
<td>25,929</td>
<td>0.22s</td>
<td>5,612</td>
<td>0.08s</td>
</tr>
<tr>
<td>5,000</td>
<td>749,826</td>
<td>372,802</td>
<td>5.62s</td>
<td>41,397</td>
<td>0.73s</td>
</tr>
<tr>
<td>7,500</td>
<td>1,684,978</td>
<td>947,908</td>
<td>13.29s</td>
<td>100,120</td>
<td>1.71s</td>
</tr>
<tr>
<td>10,000</td>
<td>2,997,625</td>
<td>1,700,163</td>
<td>28.85s</td>
<td>66,379</td>
<td>1.31s</td>
</tr>
</tbody>
</table>
# Experiments: Probabilistic

<table>
<thead>
<tr>
<th>nodes</th>
<th>edges</th>
<th>$\text{visited}_{Dij}$</th>
<th>$\text{time}_{Dij}$</th>
<th>$\text{visited}_{A^*}$</th>
<th>$\text{time}_{A^*}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>30,111</td>
<td>16,330</td>
<td>0.14s</td>
<td>902</td>
<td>0.01s</td>
</tr>
<tr>
<td>5,000</td>
<td>749,826</td>
<td>365,066</td>
<td>5.72s</td>
<td>7,607</td>
<td>0.08s</td>
</tr>
<tr>
<td>7,500</td>
<td>1,684,978</td>
<td>1,636,157</td>
<td>19.04s</td>
<td>22,250</td>
<td>0.33s</td>
</tr>
<tr>
<td>10,000</td>
<td>2,997,625</td>
<td>2,743,029</td>
<td>36.32s</td>
<td>56,021</td>
<td>1.07s</td>
</tr>
</tbody>
</table>
## Experiments: Max/Min

<table>
<thead>
<tr>
<th>nodes</th>
<th>edges</th>
<th>visited	extsubscript{Dij}</th>
<th>time	extsubscript{Dij}</th>
<th>visited	extsubscript{A*}</th>
<th>time	extsubscript{A*}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>30,111</td>
<td>24,226</td>
<td>0.24s</td>
<td>23,570</td>
<td>0.27s</td>
</tr>
<tr>
<td>5,000</td>
<td>749,826</td>
<td>600,615</td>
<td>7.69s</td>
<td>264,523</td>
<td>4.13s</td>
</tr>
<tr>
<td>7,500</td>
<td>1,684,978</td>
<td>233,162</td>
<td>4.57s</td>
<td>159,229</td>
<td>3.1s</td>
</tr>
<tr>
<td>10,000</td>
<td>2,997,625</td>
<td>1,109,862</td>
<td>23.62s</td>
<td>1,028,962</td>
<td>19.59s</td>
</tr>
</tbody>
</table>
## Shortest-quickest path

<table>
<thead>
<tr>
<th>nodes</th>
<th>edges</th>
<th>visited (_{Dij})</th>
<th>time (_{Dij})</th>
<th>visited (_{A^*})</th>
<th>time (_{A^*})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,473</td>
<td>1,669</td>
<td>1,301</td>
<td>0.02s</td>
<td>242</td>
<td>0.01s</td>
</tr>
<tr>
<td>1,777</td>
<td>1,848</td>
<td>1,427</td>
<td>0.02s</td>
<td>459</td>
<td>0.01s</td>
</tr>
<tr>
<td>2,481</td>
<td>2,609</td>
<td>1,670</td>
<td>0.02s</td>
<td>1,602</td>
<td>0.02s</td>
</tr>
<tr>
<td>54,278</td>
<td>58,655</td>
<td>44,236</td>
<td>0.17s</td>
<td>18,815</td>
<td>0.10s</td>
</tr>
</tbody>
</table>
Summary

- Formalized a general notion of cost.
- Prefix optimality.
- Generalized Dijkstra’s algorithm and A* to work on this general cost structure.
- Practical Feasibility.