



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 ?

Revision: Monoid

Let A be a set and $\times : A \times A \rightarrow A$ be a binary operator. A *monoid* is a tuple $\langle A, \times, \mathbf{1} \rangle$ if $\mathbf{1} \in A$ and for all $a, b, c \in A$

- (1) $a \times b \in A$ (closeness)
- (2) $a \times (b \times c) = (a \times b) \times c$ (associativity)
- (3) $a \times \mathbf{1} = \mathbf{1} \times a = a$ (identity)

Revision: Total Order

Let A be a set. A relation $\preceq \in A \times A$ is a *total order* whenever for all $a, b, c \in A$

- (1) $a \preceq a$ (reflexivity)
- (2) $a \preceq b \wedge b \preceq a \Rightarrow a = b$ (anti-symmetry)
- (3) $a \preceq b \wedge b \preceq c \Rightarrow a \preceq c$ (transitivity)
- (4) $a \preceq b \vee b \preceq a$ (total)

Cost Algebra

A *cost algebra* is defined as a 6-tuple $\langle A, \sqcup, \times, \preceq, \mathbf{0}, \mathbf{1} \rangle$, such that

- (1) $\langle A, \times, \mathbf{1} \rangle$ is a **monoid**
- (2) \preceq is a **total order** induced by \sqcup
- (3) $\mathbf{0} = \prod A$ (**maximum**) and $\mathbf{1} = \sqcup A$ (**minimum**)
- (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 $a, b, c \in A, c \neq \mathbf{0}$.



Examples

- **Boolean:** $\langle \{ \textit{true}, \textit{false} \}, \vee, \wedge, \Rightarrow, \textit{false}, \textit{true} \rangle$
Network/service availability.

Examples

- **Boolean:** $\langle \{true, false\}, \vee, \wedge, \Rightarrow, false, true \rangle$
Network/service availability.
- **Optimization:** $\langle \mathbb{R}^+ \cup \{+\infty\}, min, +, \leq, +\infty, 0 \rangle$
Price, propagation delay.

Examples

- **Boolean:** $\langle \{true, false\}, \vee, \wedge, \Rightarrow, false, true \rangle$
Network/service availability.
- **Optimization:** $\langle \mathbb{R}^+ \cup \{+\infty\}, min, +, \leq, +\infty, 0 \rangle$
Price, propagation delay.
- **Max/Min:** $\langle \mathbb{R}^+ \cup \{+\infty\}, max, min, \geq, 0, +\infty \rangle$
Bandwidth.

Examples

- **Boolean:** $\langle \{true, false\}, \vee, \wedge, \Rightarrow, false, true \rangle$
Network/service availability.
- **Optimization:** $\langle \mathbb{R}^+ \cup \{+\infty\}, min, +, \leq, +\infty, 0 \rangle$
Price, propagation delay.
- **Max/Min:** $\langle \mathbb{R}^+ \cup \{+\infty\}, max, min, \geq, 0, +\infty \rangle$
Bandwidth.
- **Probabilistic:** $\langle [0, 1], min, \cdot, \geq, 0, 1 \rangle$
Performance and rates.

Examples

- **Boolean:** $\langle \{true, false\}, \vee, \wedge, \Rightarrow, false, true \rangle$
Network/service availability.
- **Optimization:** $\langle \mathbb{R}^+ \cup \{+\infty\}, min, +, \leq, +\infty, 0 \rangle$
Price, propagation delay.
- **Max/Min:** $\langle \mathbb{R}^+ \cup \{+\infty\}, max, min, \geq, 0, +\infty \rangle$
Bandwidth.
- **Probabilistic:** $\langle [0, 1], min, \cdot, \geq, 0, 1 \rangle$
Performance and rates.
- **Fuzzy:** $\langle [0, 1], max, min, \geq, 0, 1 \rangle$
Performance and rates.



The Principle of Optimality

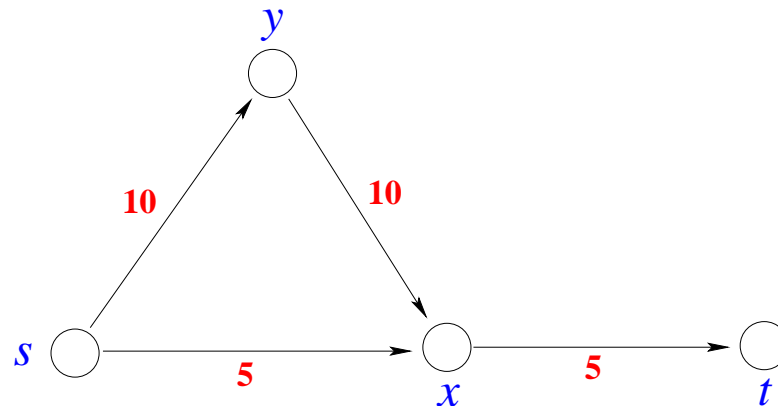
Definition: The *Principle of Optimality* requires
$$\delta(s, v) = \min \{ \delta(s, u) \times \omega(e) \mid u \xrightarrow{e} v \},$$
 where s is the start node in a given graph G .

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** = $\min(e) | e \in P$



- 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**:

$a \prec b \implies a \times c \prec b \times c$ and $c \times a \prec c \times b$ for all $a, b, c \in A$,
 $c \neq \mathbf{0}$.

Recall Max/Min: $\langle \mathbb{R}^+ \cup \{+\infty\}, \max, \min, \geq, 0, +\infty \rangle$

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



Cost-Algebraic Dijkstra

Initialization $f(u_0) \leftarrow \mathbf{1}$; $Open \leftarrow \{u_0\}$;
for each $u \neq u_0 : f(u) \leftarrow \mathbf{0}$



Cost-Algebraic Dijkstra

Initialization $f(u_0) \leftarrow \mathbf{1}$; $Open \leftarrow \{u_0\}$;
for each $u \neq u_0$: $f(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with
 $f(u) = \sqcup \{f(v) \mid v \in Open\}$

Cost-Algebraic Dijkstra

Initialization $f(u_0) \leftarrow \mathbf{1}$; $Open \leftarrow \{u_0\}$;
for each $u \neq u_0$: $f(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with
 $f(u) = \sqcup \{f(v) \mid v \in Open\}$

Update $f(v) \leftarrow$
 $\sqcup \{ \{f(v)\} \cup \{f(u) \times \omega(e) \mid u \xrightarrow{e} v\} \}$

Cost-Algebraic Dijkstra

Initialization $f(u_0) \leftarrow \mathbf{1}$; $Open \leftarrow \{u_0\}$;
for each $u \neq u_0$: $f(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with
 $f(u) = \sqcap \{f(v) \mid v \in Open\}$

Update $f(v) \leftarrow$
 $\sqcap \{ \{f(v)\} \cup \{f(u) \times \omega(e) \mid u \xrightarrow{e} v\} \}$

Proposition: Cost-algebraic Dijkstra finds prefix-optimal least-cost solution path

Cost-Algebraic A*

Initialization $f'(u_0) \leftarrow \underline{h(u_0)}$,
 $Open \leftarrow \{u_0\}$; **for each**
 $u \neq u_0 : f'(u) \leftarrow \mathbf{0}$

Cost-Algebraic A*

Initialization $f'(u_0) \leftarrow \underline{h(u_0)},$

$Open \leftarrow \{u_0\};$ **for each**

$u \neq u_0 : f'(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with

$\underline{f'(u)} = \underline{\sqcup\{f'(v) \mid v \in Open\}}$

Cost-Algebraic A*

Initialization $f'(u_0) \leftarrow \underline{h(u_0)},$

$Open \leftarrow \{u_0\};$ **for each**

$u \neq u_0 : f'(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with

$\underline{f'(u)} = \underline{\sqcup\{f'(v) \mid v \in Open\}}$

Update $f(v) \leftarrow$

$\sqcup\{\{f(v)\} \cup \{f(u) \times \omega(e) \mid u \xrightarrow{e} v\}\}$

$\underline{f'(v)} \leftarrow \underline{f(v) \times h(v)}$

Cost-Algebraic A*

Initialization $f'(u_0) \leftarrow \underline{h(u_0)},$

$Open \leftarrow \{u_0\};$ **for each**

$u \neq u_0 : f'(u) \leftarrow \mathbf{0}$

Selection Select $u \in Open$ with

$\underline{f'(u)} = \underline{\sqcup\{f'(v) \mid v \in Open\}}$

Update $f(v) \leftarrow$

$\sqcup\{\{f(v)\} \cup \{f(u) \times \omega(e) \mid u \xrightarrow{e} v\}\}$

$\underline{f'(v)} \leftarrow \underline{f(v) \times h(v)}$

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

Multi-criteria Search

The **Prioritized Cartesian Product** of

$C_1 = \langle A_1, \sqcup_1, \times_1, \preceq_1, \mathbf{0}_1, \mathbf{1}_1 \rangle$ and

$C_2 = \langle A_2, \sqcup_2, \times_2, \preceq_2, \mathbf{0}_2, \mathbf{1}_2 \rangle,$

$C_1 \times_p C_2$ is a tuple

$\langle A_1 \times A_2, \sqcup, \times, \preceq, (\mathbf{0}_1, \mathbf{0}_2), (\mathbf{1}_1, \mathbf{1}_2) \rangle,$

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 $a_1 \prec b_1 \vee (a_1 = b_1 \wedge a_2 \preceq b_2).$

Multi-criteria Search

The **Prioritized Cartesian Product** of

$C_1 = \langle A_1, \sqcup_1, \times_1, \preceq_1, \mathbf{0}_1, \mathbf{1}_1 \rangle$ and

$C_2 = \langle A_2, \sqcup_2, \times_2, \preceq_2, \mathbf{0}_2, \mathbf{1}_2 \rangle,$

$C_1 \times_p C_2$ is a tuple

$\langle A_1 \times A_2, \sqcup, \times, \preceq, (\mathbf{0}_1, \mathbf{0}_2), (\mathbf{1}_1, \mathbf{1}_2) \rangle,$

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 $a_1 \prec b_1 \vee (a_1 = b_1 \wedge a_2 \preceq b_2).$

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++.



Experiments

- Implemented generalized **Dijkstra** and **A*** in C++.
- Made possible by the **template** facility of C++.



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.



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

nodes	edges	visited _{<i>Dij</i>}	time _{<i>Dij</i>}	visited _{<i>A*</i>}	time _{<i>A*</i>}
1,000	30,111	25,929	0.22s	5,612	0.08s
5,000	749,826	372,802	5.62s	41,397	0.73s
7,500	1,684,978	947,908	13.29s	100,120	1.71s
10,000	2,997,625	1,700,163	28.85s	66,379	1.31s

Experiments: Probabilistic

nodes	edges	visited _{<i>Dij</i>}	time _{<i>Dij</i>}	visited _{<i>A*</i>}	time _{<i>A*</i>}
1,000	30,111	16,330	0.14s	902	0.01s
5,000	749,826	365,066	5.72s	7,607	0.08s
7,500	1,684,978	1,636,157	19.04s	22,250	0.33s
10,000	2,997,625	2,743,029	36.32s	56,021	1.07s

Experiments: Max/Min

nodes	edges	visited _{Dij}	time _{Dij}	visited _{A*}	time _{A*}
1,000	30,111	24,226	0.24s	23,570	0.27s
5,000	749,826	600,615	7.69s	264,523	4.13s
7,500	1,684,978	233,162	4.57s	159,229	3.1s
10,000	2,997,625	1,109,862	23.62s	1,028,962	19.59s

Shortest-quickest path

nodes	edges	visited _{Dij}	time _{Dij}	visited _{A*}	time _{A*}
1,473	1,669	1,301	0.02s	242	0.01s
1,777	1,848	1,427	0.02s	459	0.01s
2,481	2,609	1,670	0.02s	1,602	0.02s
54,278	58,655	44,236	0.17s	18,815	0.10s



Summary

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