FORMALIZED MATHEMATICS
Volume 11, Number 3, 2003
University of Białystok

Dijkstra's Shortest Path Algorithm

Jing-Chao Chen
Donghua University
Shanghai

Summary. The article formalizes Dijkstra's shortest path algorithm [11]. A path from a source vertex v to a target vertex u is said to be the shortest path if its total cost is minimum among all v-to-u paths. Dijkstra's algorithm is based on the following assumptions:

- All edge costs are non-negative.
- The number of vertices is finite.
- The source is a single vertex, but the target may be all other vertices.

The underlying principle of the algorithm may be described as follows: the algorithm starts with the source; it visits the vertices in order of increasing cost, and maintains a set V of visited vertices (denoted by UsedVx in the article) whose cost from the source has been computed, and a tentative cost $D(u)$ to each unvisited vertex u. In the article, the set of all unvisited vertices is denoted by UnusedVx. $D(u)$ is the cost of the shortest path from the source to u in the subgraph induced by $V \cup \{u\}$. We denote the set of all unvisited vertices whose D-values are not infinite (i.e. in the subgraph each of which has a path from the source to itself) by OuterVx. Dijkstra's algorithm repeatedly searches OuterVx for the vertex with minimum tentative cost (this procedure is called findmin in the article), adds it to the set V and modifies D-values by a procedure, called Relax. Suppose the unvisited vertex with minimum tentative cost is x, the procedure Relax replaces $D(u)$ with $\min\{D(u), D(u) + cost(x, u)\}$ where u is a vertex in UnusedVx, and cost(x, u) is the cost of edge (x, u). In the Mizar library, there are several computer models, e.g. SCMFSA and SCMPDS etc. However, it is extremely difficult to use these models to formalize the algorithm. Instead, we adopt functions in the Mizar library, which seem to be pseudo-codes, and are similar to those in the functional programming language, e.g. Lisp. To date, there is no rigorous justification with respect to the correctness of Dijkstra's algorithm. The article presents first the rigorous justification.

MML Identifier: `GRAPHSP`.

ISSN 1426–2630

The papers [12], [2], [20], [19], [22], [23], [6], [3], [5], [21], [1], [10], [13], [7], [15], [9], [16], [18], [8], [14], [17], and [4] provide the terminology and notation for this paper.

1. Preliminaries

For simplicity, we adopt the following rules: X denotes a set, i, j, k, m, n denote natural numbers, p denotes a finite sequence of elements of X, and i_1 denotes an integer.

We now state three propositions:

(1) For every finite sequence p and for every set x holds $x \notin \operatorname{rng} p$ and p is one-to-one iff $p \frown \langle x \rangle$ is one-to-one.

(2) If $1 \leqslant i_1$ and $i_1 \leqslant \operatorname{len} p$, then $p(i_1) \in X$.

(3) If $1 \leqslant i_1$ and $i_1 \leqslant \operatorname{len} p$, then $p_{i_1} = p(i_1)$.

For simplicity, we adopt the following rules: G denotes a graph, p_1, q_1 denote finite sequences of elements of the edges of G, p, q denote oriented chains of G, W denotes a function, U, V, e, e_1 denote sets, and v_1, v_2, v_3, v_4 denote vertices of G.

We now state three propositions:

(4) If W is weight of G and $\operatorname{len} p_1 = 1$, then $\operatorname{cost}(p_1, W) = W(p_1(1))$.

(5) If $e \in$ the edges of G, then $\langle e \rangle$ is a Simple oriented chain of G.

(6) Let p be a Simple oriented chain of G. Suppose $p = p_1 \frown q_1$ and $\operatorname{len} p_1 \geqslant 1$ and $\operatorname{len} q_1 \geqslant 1$. Then (the target of G)$(p(\operatorname{len} p)) \neq$ (the target of G)$(p_1(\operatorname{len} p_1))$ and (the source of G)$(p(1)) \neq$ (the source of G)$(q_1(1))$.

2. The Fundamental Properties of Directed Paths and Shortest Paths

We now state several propositions:

(7) p is oriented path from v_1 to v_2 in V iff p is oriented path from v_1 to v_2 in $V \cup \{v_2\}$.

(8) p is shortest path from v_1 to v_2 in V w.r.t. W iff p is shortest path from v_1 to v_2 in $V \cup \{v_2\}$ w.r.t. W.

(9) Suppose p is shortest path from v_1 to v_2 in V w.r.t. W and q is shortest path from v_1 to v_2 in V w.r.t. W. Then $\operatorname{cost}(p, W) = \operatorname{cost}(q, W)$.

(10) Let G be an oriented graph, v_1, v_2 be vertices of G, and e_2, e_3 be sets. Suppose $e_2 \in$ the edges of G and $e_3 \in$ the edges of G and e_2 orientedly joins v_1, v_2 and e_3 orientedly joins v_1, v_2. Then $e_2 = e_3$.

(11) Suppose that
(i) the vertices of $G = U \cup V$,
(ii) $v_1 \in U$,
(iii) $v_2 \in V$, and
(iv) for all v_3, v_4 such that $v_3 \in U$ and $v_4 \in V$ it is not true that there exists e such that $e \in$ the edges of G and e orientedly joins v_3, v_4.
Then there exists no p which is oriented path from v_1 to v_2.

(12) Suppose that
(i) the vertices of $G = U \cup V$,
(ii) $v_1 \in U$,
(iii) for all v_3, v_4 such that $v_3 \in U$ and $v_4 \in V$ it is not true that there exists e such that $e \in$ the edges of G and e orientedly joins v_3, v_4, and
(iv) p is oriented path from v_1 to v_2.
Then p is oriented path from v_1 to v_2 in U.

3. The Basic Theorems for Dijkstra's Shortest Path Algorithm (continue)

We adopt the following convention: G is a finite graph, P, Q are oriented chains of G, and v_1, v_2, v_3 are vertices of G.

Next we state the proposition

(13) Suppose that W is nonnegative weight of G and P is shortest path from v_1 to v_2 in V w.r.t. W and $v_1 \neq v_2$ and $v_1 \neq v_3$ and Q is shortest path from v_1 to v_3 in V w.r.t. W and it is not true that there exists e such that $e \in$ the edges of G and e orientedly joins v_2, v_3 and P is longest in shortest path from v_1 in V w.r.t. W. Then Q is shortest path from v_1 to v_3 in $V \cup \{v_2\}$ w.r.t. W.

For simplicity, we adopt the following rules: G is a finite oriented graph, P, Q are oriented chains of G, W is a function from the edges of G into $\mathbb{R}_{\geqslant 0}$, and v_1, v_2, v_3, v_4 are vertices of G.

One can prove the following three propositions:

(14) Suppose $e \in$ the edges of G and $v_1 \neq v_2$ and $P = \langle e \rangle$ and e orientedly joins v_1, v_2. Then P is shortest path from v_1 to v_2 in $\{v_1\}$ w.r.t. W.

(15) Suppose that $e \in$ the edges of G and P is shortest path from v_1 to v_2 in V w.r.t. W and $v_1 \neq v_3$ and $Q = P \frown \langle e \rangle$ and e orientedly joins v_2, v_3 and $v_1 \in V$ and for every v_4 such that $v_4 \in V$ it is not true that there exists e_1 such that $e_1 \in$ the edges of G and e_1 orientedly joins v_4, v_3. Then Q is shortest path from v_1 to v_3 in $V \cup \{v_2\}$ w.r.t. W.

(16) Suppose that
(i) the vertices of $G = U \cup V$,

(ii) $v_1 \in U$, and

(iii) for all v_3, v_4 such that $v_3 \in U$ and $v_4 \in V$ it is not true that there exists e such that $e \in$ the edges of G and e orientedly joins v_3, v_4.

Then P is shortest path from v_1 to v_2 in U w.r.t. W if and only if P is shortest path from v_1 to v_2 in W.

4. The Definition of Assignment Statement

Let f be a function and let i, x be sets. We introduce $f_i{:=}x$ as a synonym of $f +\cdot (i, x)$.

We now state the proposition

(17) For all sets x, y and for every function f holds $\operatorname{rng}(f_x{:=}y) \subseteq \operatorname{rng} f \cup \{y\}$.

Let f be a finite sequence of elements of $\mathbb{R}$, let x be a set, and let r be a real number. Then $f_x{:=}r$ is a finite sequence of elements of $\mathbb{R}$.

Let i, k be natural numbers, let f be a finite sequence of elements of $\mathbb{R}$, and let r be a real number. The functor $(f, i) := (k, r)$ yielding a finite sequence of elements of $\mathbb{R}$ is defined by:

(Def. 1) $(f, i) := (k, r) = f_i{:=}k_k{:=}r$.

In the sequel f, g, h denote elements of $\mathbb{R}^*$ and r denotes a real number.

One can prove the following propositions:

(18) If $i \neq k$ and $i \in \operatorname{dom} f$, then $((f, i) := (k, r))(i) = k$.

(19) If $m \neq i$ and $m \neq k$ and $m \in \operatorname{dom} f$, then $((f, i) := (k, r))(m) = f(m)$.

(20) If $k \in \operatorname{dom} f$, then $((f, i) := (k, r))(k) = r$.

(21) $\operatorname{dom}((f, i) := (k, r)) = \operatorname{dom} f$.

5. The Definition of Pascal–Like “while” - “do” Statement

Let X be a set. Then id_X is an element of X^X.

Let X be a set and let f, g be functions from X into X. Then $g \cdot f$ is a function from X into X.

Let X be a set and let f, g be elements of X^X. Then $g \cdot f$ is an element of X^X.

Let X be a set, let f be an element of X^X, and let g be an element of X. Then $f(g)$ is an element of X.

Let X be a set and let f be an element of X^X. The functor repeat f yields a function from $\mathbb{N}$ into X^X and is defined by:

(Def. 2) $(\operatorname{repeat} f)(0) = \mathrm{id}_X$ and for every natural number i and for every element x of X^X such that $x = (\operatorname{repeat} f)(i)$ holds $(\operatorname{repeat} f)(i + 1) = f \cdot x$.

Next we state two propositions:

(22) For every element F of $(\mathbb{R}^*)^{\mathbb{R}^*}$ and for every element f of $\mathbb{R}^*$ and for all natural numbers n, i holds $(\text{repeat}\,F)(0)(f) = f$.

(23) Let F, G be elements of $(\mathbb{R}^*)^{\mathbb{R}^*}$, f be an element of $\mathbb{R}^*$, and i be a natural number. Then $(\text{repeat}(F \cdot G))(i+1)(f) = F(G((\text{repeat}(F \cdot G))(i)(f)))$.

Let g be an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ and let f be an element of $\mathbb{R}^*$. Then $g(f)$ is an element of $\mathbb{R}^*$.

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. The functor $\text{OuterVx}(f, n)$ yielding a subset of $\mathbb{N}$ is defined by:

(Def. 3) $\text{OuterVx}(f, n) = \{i : i \in \text{dom}\, f \wedge 1 \leqslant i \wedge i \leqslant n \wedge f(i) \neq -1 \wedge f(n+i) \neq -1\}$.

Let f be an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$, let g be an element of $\mathbb{R}^*$, and let n be a natural number. Let us assume that there exists i such that $\text{OuterVx}((\text{repeat}\,f)(i)(g), n) = \emptyset$. The functor $\text{LifeSpan}(f, g, n)$ yielding a natural number is defined by:

(Def. 4) $\text{OuterVx}((\text{repeat}\,f)(\text{LifeSpan}(f, g, n))(g), n) = \emptyset$ and for every natural number k such that $\text{OuterVx}((\text{repeat}\,f)(k)(g), n) = \emptyset$ holds $\text{LifeSpan}(f, g, n) \leqslant k$.

Let f be an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ and let n be a natural number. The functor $\text{WhileDo}(f, n)$ yielding an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ is defined as follows:

(Def. 5) $\text{dom}\,\text{WhileDo}(f, n) = \mathbb{R}^*$ and for every element h of $\mathbb{R}^*$ holds $(\text{WhileDo}(f, n))(h) = (\text{repeat}\,f)(\text{LifeSpan}(f, h, n))(h)$.

6. Defining a Weight Function for an Oriented Graph

Let G be an oriented graph and let v_1, v_2 be vertices of G. Let us assume that there exists a set e such that $e \in$ the edges of G and e orientedly joins v_1, v_2. The functor $\text{Edge}(v_1, v_2)$ is defined as follows:

(Def. 6) There exists a set e such that $\text{Edge}(v_1, v_2) = e$ and $e \in$ the edges of G and e orientedly joins v_1, v_2.

Let G be an oriented graph, let v_1, v_2 be vertices of G, and let W be a function. The functor $\text{Weight}(v_1, v_2, W)$ is defined as follows:

(Def. 7) $$\text{Weight}(v_1, v_2, W) = \begin{cases} W(\text{Edge}(v_1, v_2)), & \text{if there exists a set } e \text{ such that } e \in \text{the edges of } G \text{ and } e \text{ orientedly joins } v_1,\ v_2, \\ -1, & \text{otherwise.} \end{cases}$$

Let G be an oriented graph, let v_1, v_2 be vertices of G, and let W be a function from the edges of G into $\mathbb{R}_{\geqslant 0}$. Then $\text{Weight}(v_1, v_2, W)$ is a real number.

In the sequel G is an oriented graph, v_1, v_2 are vertices of G, and W is a function from the edges of G into $\mathbb{R}_{\geqslant 0}$.

We now state three propositions:

(24) Weight$(v_1, v_2, W) \geqslant 0$ iff there exists a set e such that $e \in$ the edges of G and e orientedly joins v_1, v_2.

(25) Weight$(v_1, v_2, W) = -1$ iff it is not true that there exists a set e such that $e \in$ the edges of G and e orientedly joins v_1, v_2.

(26) If $e \in$ the edges of G and e orientedly joins v_1, v_2, then Weight$(v_1, v_2, W) = W(e)$.

7. Basic Operations for Dijkstra's Shortest Path Algorithm

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. The functor UnusedVx(f, n) yields a subset of $\mathbb{N}$ and is defined as follows:

(Def. 8) UnusedVx$(f, n) = \{i : i \in \mathrm{dom}\, f \ \wedge\ 1 \leqslant i \ \wedge\ i \leqslant n \ \wedge\ f(i) \neq -1\}$.

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. The functor UsedVx(f, n) yielding a subset of $\mathbb{N}$ is defined as follows:

(Def. 9) UsedVx$(f, n) = \{i : i \in \mathrm{dom}\, f \ \wedge\ 1 \leqslant i \ \wedge\ i \leqslant n \ \wedge\ f(i) = -1\}$.

The following proposition is true

(27) UnusedVx$(f, n) \subseteq \mathrm{Seg}\, n$.

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. One can verify that UnusedVx(f, n) is finite.

Next we state two propositions:

(28) OuterVx$(f, n) \subseteq$ UnusedVx(f, n).

(29) OuterVx$(f, n) \subseteq \mathrm{Seg}\, n$.

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. Observe that OuterVx(f, n) is finite.

Let X be a finite subset of $\mathbb{N}$, let f be an element of $\mathbb{R}^*$, and let us consider n. The functor Argmin(X, f, n) yielding a natural number is defined by the conditions (Def. 10).

(Def. 10)(i) If $X \neq \emptyset$, then there exists i such that $i =$ Argmin(X, f, n) and $i \in X$ and for every k such that $k \in X$ holds $f_{2 \cdot n+i} \leqslant f_{2 \cdot n+k}$ and for every k such that $k \in X$ and $f_{2 \cdot n+i} = f_{2 \cdot n+k}$ holds $i \leqslant k$, and

(ii) if $X = \emptyset$, then Argmin$(X, f, n) = 0$.

We now state two propositions:

(30) If OuterVx$(f, n) \neq \emptyset$ and $j =$ Argmin(OuterVx$(f, n), f, n)$, then $j \in \mathrm{dom}\, f$ and $1 \leqslant j$ and $j \leqslant n$ and $f(j) \neq -1$ and $f(n + j) \neq -1$.

(31) Argmin(OuterVx$(f, n), f, n) \leqslant n$.

Let n be a natural number. The functor findmin n yields an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ and is defined as follows:

(Def. 11) dom findmin $n = \mathbb{R}^*$ and for every element f of $\mathbb{R}^*$ holds (findmin $n)(f) = (f, n \cdot n + 3 \cdot n + 1) :=$ (Argmin(OuterVx$(f, n), f, n), -1)$.

Next we state four propositions:

(32) If $i \in \operatorname{dom} f$ and $i > n$ and $i \neq n \cdot n + 3 \cdot n + 1$, then $(\operatorname{findmin} n)(f)(i) = f(i)$.

(33) If $i \in \operatorname{dom} f$ and $f(i) = -1$ and $i \neq n{\cdot}n{+}3{\cdot}n{+}1$, then $(\operatorname{findmin} n)(f)(i) = -1$.

(34) $\operatorname{dom}(\operatorname{findmin} n)(f) = \operatorname{dom} f$.

(35) If $\operatorname{OuterVx}(f, n) \neq \emptyset$, then there exists j such that $j \in \operatorname{OuterVx}(f, n)$ and $1 \leqslant j$ and $j \leqslant n$ and $(\operatorname{findmin} n)(f)(j) = -1$.

Let f be an element of $\mathbb{R}^*$ and let n, k be natural numbers. The functor newpathcost(f, n, k) yielding a real number is defined as follows:

(Def. 12) $\operatorname{newpathcost}(f, n, k) = f_{2 \cdot n + f_{n \cdot n + 3 \cdot n + 1}} + f_{2 \cdot n + n \cdot f_{n \cdot n + 3 \cdot n + 1} + k}$.

Let n, k be natural numbers and let f be an element of $\mathbb{R}^*$. We say that f has better path at n, k if and only if:

(Def. 13) $f(n+k) = -1$ or $f_{2 \cdot n + k} > \operatorname{newpathcost}(f, n, k)$ but $f_{2 \cdot n + n \cdot f_{n \cdot n + 3 \cdot n + 1} + k} \geqslant 0$ but $f(k) \neq -1$.

Let f be an element of $\mathbb{R}^*$ and let n be a natural number. The functor Relax(f, n) yields an element of $\mathbb{R}^*$ and is defined by the conditions (Def. 14).

(Def. 14)(i) $\operatorname{dom} \operatorname{Relax}(f, n) = \operatorname{dom} f$, and

(ii) for every natural number k such that $k \in \operatorname{dom} f$ holds if $n < k$ and $k \leqslant 2 \cdot n$, then if f has better path at n, $k -' n$, then $(\operatorname{Relax}(f, n))(k) = f_{n \cdot n + 3 \cdot n + 1}$ and if f does not have better path at n, $k -' n$, then $(\operatorname{Relax}(f, n))(k) = f(k)$ and if $2 \cdot n < k$ and $k \leqslant 3 \cdot n$, then if f has better path at n, $k -' 2 \cdot n$, then $(\operatorname{Relax}(f, n))(k) = \operatorname{newpathcost}(f, n, k -' 2 \cdot n)$ and if f does not have better path at n, $k -' 2 \cdot n$, then $(\operatorname{Relax}(f, n))(k) = f(k)$ and if $k \leqslant n$ or $k > 3 \cdot n$, then $(\operatorname{Relax}(f, n))(k) = f(k)$.

Let n be a natural number. The functor Relax n yields an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ and is defined by:

(Def. 15) $\operatorname{dom} \operatorname{Relax} n = \mathbb{R}^*$ and for every element f of $\mathbb{R}^*$ holds $(\operatorname{Relax} n)(f) = \operatorname{Relax}(f, n)$.

One can prove the following propositions:

(36) $\operatorname{dom}(\operatorname{Relax} n)(f) = \operatorname{dom} f$.

(37) If $i \leqslant n$ or $i > 3 \cdot n$ and if $i \in \operatorname{dom} f$, then $(\operatorname{Relax} n)(f)(i) = f(i)$.

(38) $\operatorname{dom}(\operatorname{repeat}(\operatorname{Relax} n{\cdot}\operatorname{findmin} n))(i)(f) = \operatorname{dom}(\operatorname{repeat}(\operatorname{Relax} n{\cdot}\operatorname{findmin} n))(i+1)(f)$.

(39) If $\operatorname{OuterVx}((\operatorname{repeat}(\operatorname{Relax} n \cdot \operatorname{findmin} n))(i)(f), n) \neq \emptyset$, then $\operatorname{UnusedVx}((\operatorname{repeat}(\operatorname{Relax} n \cdot \operatorname{findmin} n))(i+1)(f), n) \subset \operatorname{UnusedVx}((\operatorname{repeat}(\operatorname{Relax} n \cdot \operatorname{findmin} n))(i)(f), n)$.

(40) If $g = (\operatorname{repeat}(\operatorname{Relax} n \cdot \operatorname{findmin} n))(i)(f)$ and $h = (\operatorname{repeat}(\operatorname{Relax} n \cdot \operatorname{findmin} n))(i + 1)(f)$ and $k = \operatorname{Argmin}(\operatorname{OuterVx}(g, n), g, n)$ and

OuterVx$(g, n) \neq \emptyset$, then UsedVx$(h, n) =$ UsedVx$(g, n) \cup \{k\}$ and $k \notin$ UsedVx(g, n).

(41) There exists i such that $i \leqslant n$ and OuterVx((repeat(Relax $n \cdot$ findmin n))$(i)(f), n) = \emptyset$.

(42) dom $f =$ dom(repeat(Relax $n \cdot$ findmin n))$(i)(f)$.

Let f, g be elements of $\mathbb{R}^*$ and let us consider m, n. We say that f, g are equal at m, n if and only if:

(Def. 16) dom $f =$ dom g and for every k such that $k \in$ dom f and $m \leqslant k$ and $k \leqslant n$ holds $f(k) = g(k)$.

One can prove the following propositions:

(43) f, f are equal at m, n.

(44) If f, g are equal at m, n and g, h are equal at m, n, then f, h are equal at m, n.

(45) (repeat(Relax $n \cdot$ findmin n))$(i)(f)$, (repeat(Relax $n \cdot$ findmin n))$(i+1)(f)$ are equal at $3 \cdot n + 1$, $n \cdot n + 3 \cdot n$.

(46) Let F be an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$, f be an element of $\mathbb{R}^*$, and n, i be natural numbers. If $i <$ LifeSpan(F, f, n), then OuterVx((repeat F)$(i)(f), n) \neq \emptyset$.

(47) f, (repeat(Relax $n \cdot$ findmin n))$(i)(f)$ are equal at $3 \cdot n + 1$, $n \cdot n + 3 \cdot n$.

(48) Suppose that

(i) $1 \leqslant n$,

(ii) $1 \in$ dom f,

(iii) $f(n+1) \neq -1$,

(iv) for every i such that $1 \leqslant i$ and $i \leqslant n$ holds $f(i) = 1$, and

(v) for every i such that $2 \leqslant i$ and $i \leqslant n$ holds $f(n+i) = -1$.

Then $1 =$ Argmin(OuterVx$(f, n), f, n)$ and UsedVx$(f, n) = \emptyset$ and $\{1\} =$ UsedVx((repeat(Relax $n \cdot$ findmin n))$(1)(f), n)$.

(49) If $g =$ (repeat(Relax $n \cdot$ findmin n))$(1)(f)$ and $h =$ (repeat(Relax $n \cdot$ findmin n))$(i)(f)$ and $1 \leqslant i$ and $i \leqslant$ LifeSpan(Relax $n \cdot$ findmin $n, f, n)$ and $m \in$ UsedVx(g, n), then $m \in$ UsedVx(h, n).

Let p be a finite sequence of elements of $\mathbb{N}$, let f be an element of $\mathbb{R}^*$, and let i, n be natural numbers. We say that p is vertex sequence at f, i, n if and only if:

(Def. 17) $p(\mathrm{len}\, p) = i$ and for every k such that $1 \leqslant k$ and $k < \mathrm{len}\, p$ holds $p(\mathrm{len}\, p - k) = f(n + p_{(\mathrm{len}\, p - k)+1})$.

Let p be a finite sequence of elements of $\mathbb{N}$, let f be an element of $\mathbb{R}^*$, and let i, n be natural numbers. We say that p is simple vertex sequence at f, i, n if and only if:

(Def. 18) $p(1) = 1$ and $\mathrm{len}\, p > 1$ and p is vertex sequence at f, i, n and one-to-one.

Next we state the proposition

(50) Let p, q be finite sequences of elements of $\mathbb{N}$, f be an element of $\mathbb{R}^*$, and i, n be natural numbers. Suppose p is simple vertex sequence at f, i, n and q is simple vertex sequence at f, i, n. Then $p = q$.

Let G be a graph, let p be a finite sequence of elements of the edges of G, and let v_5 be a finite sequence. We say that p is oriented edge sequence at v_5 if and only if:

(Def. 19) $\operatorname{len} v_5 = \operatorname{len} p + 1$ and for every n such that $1 \leqslant n$ and $n \leqslant \operatorname{len} p$ holds (the source of G)$(p(n)) = v_5(n)$ and (the target of G)$(p(n)) = v_5(n+1)$.

One can prove the following two propositions:

(51) Let G be an oriented graph, v_5 be a finite sequence, and p, q be oriented chains of G. Suppose p is oriented edge sequence at v_5 and q is oriented edge sequence at v_5. Then $p = q$.

(52) Let G be a graph, v_6, v_7 be finite sequences, and p be an oriented chain of G. Suppose p is oriented edge sequence at v_6 and oriented edge sequence at v_7 and $\operatorname{len} p \geqslant 1$. Then $v_6 = v_7$.

8. Data Structure for Dijkstra's Shortest Path Algorithm

Let f be an element of $\mathbb{R}^*$, let G be an oriented graph, let n be a natural number, and let W be a function from the edges of G into $\mathbb{R}_{\geqslant 0}$. We say that f is input of Dijkstra algorithm G to n in W if and only if the conditions (Def. 20) are satisfied.

(Def. 20)(i) $\operatorname{len} f = n \cdot n + 3 \cdot n + 1$,

(ii) $\operatorname{Seg} n =$ the vertices of G,

(iii) for every i such that $1 \leqslant i$ and $i \leqslant n$ holds $f(i) = 1$ and $f(2 \cdot n + i) = 0$,

(iv) $f(n+1) = 0$,

(v) for every i such that $2 \leqslant i$ and $i \leqslant n$ holds $f(n+i) = -1$, and

(vi) for all vertices i, j of G and for all k, m such that $k = i$ and $m = j$ holds $f(2 \cdot n + n \cdot k + m) = \operatorname{Weight}(i, j, W)$.

9. The Definition of Dijkstra's Shortest Path Algorithm

Let n be a natural number. The functor $\operatorname{DijkstraAlgorithm} n$ yielding an element of $(\mathbb{R}^*)^{\mathbb{R}^*}$ is defined as follows:

(Def. 21) $\operatorname{DijkstraAlgorithm} n = \operatorname{WhileDo}(\operatorname{Relax} n \cdot \operatorname{findmin} n, n)$.

10. Justifying the Correctness of Dijkstra's Shortest Path Algorithm

For simplicity, we adopt the following rules: p is a finite sequence of elements of $\mathbb{N}$, G is a finite oriented graph, P, Q are oriented chains of G, W is a function from the edges of G into $\mathbb{R}_{\geqslant 0}$, and v_1, v_2 are vertices of G.

We now state the proposition

(53) Suppose f is input of Dijkstra algorithm G to n in W and $v_1 = 1$ and $1 \neq v_2$ and $v_2 = i$ and $n \geqslant 1$ and $g = (\text{DijkstraAlgorithm}\, n)(f)$. Then

(i) the vertices of $G = \text{UsedVx}(g, n) \cup \text{UnusedVx}(g, n)$,

(ii) if $v_2 \in \text{UsedVx}(g, n)$, then there exist p, P such that p is simple vertex sequence at g, i, n and P is oriented edge sequence at p and shortest path from v_1 to v_2 in W and $\text{cost}(P, W) = g(2 \cdot n + i)$, and

(iii) if $v_2 \in \text{UnusedVx}(g, n)$, then there exists no Q which is oriented path from v_1 to v_2.

References

[1] Grzegorz Bancerek. Cardinal numbers. *Formalized Mathematics*, 1(**2**):377–382, 1990.
[2] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.
[3] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.
[4] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Formalized Mathematics*, 5(**4**):485–492, 1996.
[5] Czesław Byliński. Finite sequences and tuples of elements of a non-empty sets. *Formalized Mathematics*, 1(**3**):529–536, 1990.
[6] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[7] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.
[8] Czesław Byliński. The sum and product of finite sequences of real numbers. *Formalized Mathematics*, 1(**4**):661–668, 1990.
[9] Jing-Chao Chen and Yatsuka Nakamura. The underlying principle of Dijkstra's shortest path algorithm. *Formalized Mathematics*, 11(**2**):143–152, 2003.
[10] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[11] E. W. Dijkstra. A note on two problems in connection with graphs. *Numer. Math.*, 1:269–271, 1959.
[12] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.
[13] Krzysztof Hryniewiecki. Graphs. *Formalized Mathematics*, 2(**3**):365–370, 1991.
[14] Jarosław Kotowicz and Yatsuka Nakamura. Introduction to Go-board - part I. *Formalized Mathematics*, 3(**1**):107–115, 1992.
[15] Yatsuka Nakamura and Piotr Rudnicki. Oriented chains. *Formalized Mathematics*, 7(**2**):189–192, 1998.
[16] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Formalized Mathematics*, 4(**1**):83–86, 1993.
[17] Andrzej Trybulec. Introduction to arithmetics. *To appear in Formalized Mathematics*.
[18] Andrzej Trybulec. Domains and their Cartesian products. *Formalized Mathematics*, 1(**1**):115–122, 1990.
[19] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[20] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.

[21] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.
[22] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[23] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received March 17, 2003