FORMALIZED MATHEMATICS
Volume 13, Number 2, Pages 253–269
University of Białystok, 2005

Walks in Graphs[1]

Gilbert Lee[2]
University of Victoria, Victoria, Canada

Summary. We define walks for graphs introduced in [9], introduce walk attributes and functors for walk creation and modification of walks. Subwalks of a walk are also defined. In our rendition, walks are alternating finite sequences of vertices and edges.

MML identifier: `GLIB_001`, version: `7.5.01 4.39.921`

The notation and terminology used here are introduced in the following papers: [14], [12], [16], [13], [18], [6], [4], [5], [1], [10], [17], [7], [3], [19], [15], [8], [2], [9], and [11].

1. Preliminaries

The following propositions are true:

(1) For all odd natural numbers x, y holds $x < y$ iff $x + 2 \leq y$.

(2) Let X be a set and k be a natural number. Suppose $X \subseteq \operatorname{Seg} k$. Let m, n be natural numbers. If $m \in \operatorname{dom} \operatorname{Sgm} X$ and $n = (\operatorname{Sgm} X)(m)$, then $m \leq n$.

(3) For every set X and for every finite sequence f_2 of elements of X and for every FinSubsequence f_1 of f_2 holds $\operatorname{len} \operatorname{Seq} f_1 \leq \operatorname{len} f_2$.

(4) Let X be a set, f_2 be a finite sequence of elements of X, f_1 be a FinSubsequence of f_2, and m be a natural number. Suppose $m \in \operatorname{dom} \operatorname{Seq} f_1$. Then there exists a natural number n such that $n \in \operatorname{dom} f_2$ and $m \leq n$ and $(\operatorname{Seq} f_1)(m) = f_2(n)$.

[1]This work has been partially supported by NSERC, Alberta Ingenuity Fund and iCORE.
[2]Part of author's MSc work.

ISSN 1426–2630

(5) For every set X and for every finite sequence f_2 of elements of X and for every FinSubsequence f_1 of f_2 holds $\operatorname{len}\operatorname{Seq} f_1 = \operatorname{card} f_1$.

(6) Let X be a set, f_2 be a finite sequence of elements of X, and f_1 be a FinSubsequence of f_2. Then $\operatorname{dom}\operatorname{Seq} f_1 = \operatorname{dom}\operatorname{Sgm}\operatorname{dom} f_1$.

2. Walk Definitions

Let G be a graph. A finite sequence of elements of the vertices of G is said to be a vertex sequence of G if:

(Def. 1) For every natural number n such that $1 \leq n$ and $n < \operatorname{len}\mathrm{it}$ there exists a set e such that e joins $\mathrm{it}(n)$ and $\mathrm{it}(n+1)$ in G.

Let G be a graph. A finite sequence of elements of the edges of G is said to be a edge sequence of G if it satisfies the condition (Def. 2).

(Def. 2) There exists a finite sequence v_1 of elements of the vertices of G such that $\operatorname{len} v_1 = \operatorname{len}\mathrm{it} + 1$ and for every natural number n such that $1 \leq n$ and $n \leq \operatorname{len}\mathrm{it}$ holds $\mathrm{it}(n)$ joins $v_1(n)$ and $v_1(n+1)$ in G.

Let G be a graph. A finite sequence of elements of (the vertices of G) $\cup$ (the edges of G) is said to be a walk of G if it satisfies the conditions (Def. 3).

(Def. 3)(i) $\operatorname{len}\mathrm{it}$ is odd,

(ii) $\mathrm{it}(1) \in$ the vertices of G, and

(iii) for every odd natural number n such that $n < \operatorname{len}\mathrm{it}$ holds $\mathrm{it}(n+1)$ joins $\mathrm{it}(n)$ and $\mathrm{it}(n+2)$ in G.

Let G be a graph and let W be a walk of G. One can verify that $\operatorname{len} W$ is odd and non empty.

Let G be a graph and let v be a vertex of G. The functor $G.\mathrm{walkOf}(v)$ yielding a walk of G is defined as follows:

(Def. 4) $G.\mathrm{walkOf}(v) = \langle v \rangle$.

Let G be a graph and let x, y, e be sets. The functor $G.\mathrm{walkOf}(x, e, y)$ yielding a walk of G is defined as follows:

(Def. 5) $G.\mathrm{walkOf}(x, e, y) = \begin{cases} \langle x, e, y \rangle, & \text{if } e \text{ joins } x \text{ and } y \text{ in } G, \\ G.\mathrm{walkOf}(\mathrm{choose}(\text{the vertices of } G)), & \text{otherwise.} \end{cases}$

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{first}()$ yields a vertex of G and is defined as follows:

(Def. 6) $W.\mathrm{first}() = W(1)$.

The functor $W.\mathrm{last}()$ yields a vertex of G and is defined by:

(Def. 7) $W.\mathrm{last}() = W(\operatorname{len} W)$.

Let G be a graph, let W be a walk of G, and let n be a natural number. The functor $W.\mathrm{vertexAt}(n)$ yielding a vertex of G is defined as follows:

(Def. 8) $W.\mathrm{vertexAt}(n) = \begin{cases} W(n), & \text{if } n \text{ is odd and } n \leq \mathrm{len}\, W, \\ W.\mathrm{first}(), & \text{otherwise.} \end{cases}$

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{reverse}()$ yielding a walk of G is defined as follows:

(Def. 9) $W.\mathrm{reverse}() = \mathrm{Rev}(W)$.

Let G be a graph and let W_1, W_2 be walks of G. The functor $W_1.\mathrm{append}(W_2)$ yields a walk of G and is defined by:

(Def. 10) $W_1.\mathrm{append}(W_2) = \begin{cases} W_1 \frown W_2, & \text{if } W_1.\mathrm{last}() = W_2.\mathrm{first}(), \\ W_1, & \text{otherwise.} \end{cases}$

Let G be a graph, let W be a walk of G, and let m, n be natural numbers. The functor $W.\mathrm{cut}(m,n)$ yields a walk of G and is defined by:

(Def. 11) $W.\mathrm{cut}(m,n) = \begin{cases} \langle W(m), \ldots, W(n) \rangle, & \text{if } m \text{ is odd and } n \text{ is odd and} \\ \quad m \leq n \text{ and } n \leq \mathrm{len}\, W, \\ W, & \text{otherwise.} \end{cases}$

Let G be a graph, let W be a walk of G, and let m, n be natural numbers. The functor $W.\mathrm{remove}(m,n)$ yielding a walk of G is defined by:

(Def. 12) $W.\mathrm{remove}(m,n) = \begin{cases} (W.\mathrm{cut}(1,m)).\mathrm{append}((W.\mathrm{cut}(n, \mathrm{len}\, W))), \\ \quad \text{if } m \text{ is odd and } n \text{ is odd and } m \leq n \text{ and} \\ \quad n \leq \mathrm{len}\, W \text{ and } W(m) = W(n), \\ W, & \text{otherwise.} \end{cases}$

Let G be a graph, let W be a walk of G, and let e be a set. The functor $W.\mathrm{addEdge}(e)$ yields a walk of G and is defined as follows:

(Def. 13) $W.\mathrm{addEdge}(e) = W.\mathrm{append}((G.\mathrm{walkOf}(W.\mathrm{last}(), e, W.\mathrm{last}().\mathrm{adj}(e))))$.

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{vertexSeq}()$ yielding a vertex sequence of G is defined by:

(Def. 14) $\mathrm{len}\, W + 1 = 2 \cdot \mathrm{len}(W.\mathrm{vertexSeq}())$ and for every natural number n such that $1 \leq n$ and $n \leq \mathrm{len}(W.\mathrm{vertexSeq}())$ holds $W.\mathrm{vertexSeq}()(n) = W(2 \cdot n - 1)$.

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{edgeSeq}()$ yields a edge sequence of G and is defined by:

(Def. 15) $\mathrm{len}\, W = 2 \cdot \mathrm{len}(W.\mathrm{edgeSeq}()) + 1$ and for every natural number n such that $1 \leq n$ and $n \leq \mathrm{len}(W.\mathrm{edgeSeq}())$ holds $W.\mathrm{edgeSeq}()(n) = W(2 \cdot n)$.

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{vertices}()$ yields a finite subset of the vertices of G and is defined as follows:

(Def. 16) $W.\mathrm{vertices}() = \mathrm{rng}(W.\mathrm{vertexSeq}())$.

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{edges}()$ yields a finite subset of the edges of G and is defined by:

(Def. 17) $W.\mathrm{edges}() = \mathrm{rng}(W.\mathrm{edgeSeq}())$.

Let G be a graph and let W be a walk of G. The functor $W.\mathrm{length}()$ yielding a natural number is defined by:

(Def. 18) W.length() = len(W.edgeSeq()).

Let G be a graph, let W be a walk of G, and let v be a set. The functor W.find(v) yields an odd natural number and is defined by:

(Def. 19)(i) W.find(v) $\leq$ len W and W(W.find(v)) $= v$ and for every odd natural number n such that $n \leq$ len W and $W(n) = v$ holds W.find(v) $\leq n$ if $v \in W$.vertices(),

(ii) W.find(v) = len W, otherwise.

Let G be a graph, let W be a walk of G, and let n be a natural number. The functor W.find(n) yielding an odd natural number is defined by:

(Def. 20)(i) W.find(n) $\leq$ len W and W(W.find(n)) $= W(n)$ and for every odd natural number k such that $k \leq$ len W and $W(k) = W(n)$ holds W.find(n) $\leq k$ if n is odd and $n \leq$ len W,

(ii) W.find(n) = len W, otherwise.

Let G be a graph, let W be a walk of G, and let v be a set. The functor W.rfind(v) yields an odd natural number and is defined as follows:

(Def. 21)(i) W.rfind(v) $\leq$ len W and W(W.rfind(v)) $= v$ and for every odd natural number n such that $n \leq$ len W and $W(n) = v$ holds $n \leq W$.rfind(v) if $v \in W$.vertices(),

(ii) W.rfind(v) = len W, otherwise.

Let G be a graph, let W be a walk of G, and let n be a natural number. The functor W.rfind(n) yields an odd natural number and is defined by:

(Def. 22)(i) W.rfind(n) $\leq$ len W and W(W.rfind(n)) $= W(n)$ and for every odd natural number k such that $k \leq$ len W and $W(k) = W(n)$ holds $k \leq W$.rfind(n) if n is odd and $n \leq$ len W,

(ii) W.rfind(n) = len W, otherwise.

Let G be a graph, let u, v be sets, and let W be a walk of G. We say that W is walk from u to v if and only if:

(Def. 23) W.first() $= u$ and W.last() $= v$.

Let G be a graph and let W be a walk of G. We say that W is closed if and only if:

(Def. 24) W.first() = W.last().

We say that W is directed if and only if:

(Def. 25) For every odd natural number n such that $n <$ len W holds (the source of G)($W(n+1)) = W(n)$.

We say that W is trivial if and only if:

(Def. 26) W.length() = 0.

We say that W is trail-like if and only if:

(Def. 27) W.edgeSeq() is one-to-one.

Let G be a graph and let W be a walk of G. We introduce W is open as an antonym of W is closed.

Let G be a graph and let W be a walk of G. We say that W is path-like if and only if the conditions (Def. 28) are satisfied.

(Def. 28)(i) W is trail-like, and

(ii) for all odd natural numbers m, n such that $m < n$ and $n \leq \operatorname{len} W$ holds if $W(m) = W(n)$, then $m = 1$ and $n = \operatorname{len} W$.

Let G be a graph and let W be a walk of G. We say that W is vertex-distinct if and only if:

(Def. 29) For all odd natural numbers m, n such that $m \leq \operatorname{len} W$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ holds $m = n$.

Let G be a graph and let W be a walk of G. We say that W is circuit-like if and only if:

(Def. 30) W is closed, trail-like, and non trivial.

We say that W is cycle-like if and only if:

(Def. 31) W is closed, path-like, and non trivial.

Let G be a graph. One can verify the following observations:

* every walk of G which is path-like is also trail-like,
* every walk of G which is trivial is also path-like,
* every walk of G which is trivial is also vertex-distinct,
* every walk of G which is vertex-distinct is also path-like,
* every walk of G which is circuit-like is also closed, trail-like, and non trivial, and
* every walk of G which is cycle-like is also closed, path-like, and non trivial.

Let G be a graph. Observe that there exists a walk of G which is closed, directed, and trivial.

Let G be a graph. Observe that there exists a walk of G which is vertex-distinct.

Let G be a graph. A trail of G is a trail-like walk of G. A path of G is a path-like walk of G.

Let G be a graph. A dwalk of G is a directed walk of G. A dtrail of G is a directed trail of G. A dpath of G is a directed path of G.

Let G be a graph and let v be a vertex of G. Note that $G.\mathrm{walkOf}(v)$ is closed, directed, and trivial.

Let G be a graph and let x, e, y be sets. One can check that $G.\mathrm{walkOf}(x, e, y)$ is path-like.

Let G be a graph and let x, e be sets. Note that $G.\mathrm{walkOf}(x, e, x)$ is closed.

Let G be a graph and let W be a closed walk of G. One can check that $W.\mathrm{reverse}()$ is closed.

Let G be a graph and let W be a trivial walk of G. One can verify that W.reverse() is trivial.

Let G be a graph and let W be a trail of G. Note that W.reverse() is trail-like.

Let G be a graph and let W be a path of G. Observe that W.reverse() is path-like.

Let G be a graph and let W_1, W_2 be closed walks of G. Note that W_1.append(W_2) is closed.

Let G be a graph and let W_1, W_2 be dwalks of G. One can verify that W_1.append(W_2) is directed.

Let G be a graph and let W_1, W_2 be trivial walks of G. Observe that W_1.append(W_2) is trivial.

Let G be a graph, let W be a dwalk of G, and let m, n be natural numbers. Note that W.cut(m, n) is directed.

Let G be a graph, let W be a trivial walk of G, and let m, n be natural numbers. Observe that W.cut(m, n) is trivial.

Let G be a graph, let W be a trail of G, and let m, n be natural numbers. Note that W.cut(m, n) is trail-like.

Let G be a graph, let W be a path of G, and let m, n be natural numbers. Note that W.cut(m, n) is path-like.

Let G be a graph, let W be a vertex-distinct walk of G, and let m, n be natural numbers. One can verify that W.cut(m, n) is vertex-distinct.

Let G be a graph, let W be a closed walk of G, and let m, n be natural numbers. One can verify that W.remove(m, n) is closed.

Let G be a graph, let W be a dwalk of G, and let m, n be natural numbers. Note that W.remove(m, n) is directed.

Let G be a graph, let W be a trivial walk of G, and let m, n be natural numbers. One can check that W.remove(m, n) is trivial.

Let G be a graph, let W be a trail of G, and let m, n be natural numbers. Observe that W.remove(m, n) is trail-like.

Let G be a graph, let W be a path of G, and let m, n be natural numbers. Observe that W.remove(m, n) is path-like.

Let G be a graph and let W be a walk of G. A walk of G is called a subwalk of W if:

(Def. 32) It is walk from W.first() to W.last() and there exists a FinSubsequence e_1 of W.edgeSeq() such that it.edgeSeq() $=$ Seq e_1.

Let G be a graph, let W be a walk of G, and let m, n be natural numbers. Then W.remove(m, n) is a subwalk of W.

Let G be a graph and let W be a walk of G. Note that there exists a subwalk of W which is trail-like and path-like.

Let G be a graph and let W be a walk of G. A trail of W is a trail-like subwalk of W. A path of W is a path-like subwalk of W.

Let G be a graph and let W be a dwalk of G. One can verify that there exists a path of W which is directed.

Let G be a graph and let W be a dwalk of G. A dwalk of W is a directed subwalk of W. A dtrail of W is a directed trail of W. A dpath of W is a directed path of W.

Let G be a graph. The functor G.allWalks() yields a non empty subset of ((the vertices of G) $\cup$ (the edges of G))* and is defined by:

(Def. 33) G.allWalks() $= \{W : W$ ranges over walks of $G\}$.

Let G be a graph. The functor G.allTrails() yielding a non empty subset of G.allWalks() is defined by:

(Def. 34) G.allTrails() $= \{W : W$ ranges over trails of $G\}$.

Let G be a graph. The functor G.allPaths() yields a non empty subset of G.allTrails() and is defined as follows:

(Def. 35) G.allPaths() $= \{W : W$ ranges over paths of $G\}$.

Let G be a graph. The functor G.allDWalks() yields a non empty subset of G.allWalks() and is defined by:

(Def. 36) G.allDWalks() $= \{W : W$ ranges over dwalks of $G\}$.

Let G be a graph. The functor G.allDTrails() yields a non empty subset of G.allTrails() and is defined as follows:

(Def. 37) G.allDTrails() $= \{W : W$ ranges over dtrails of $G\}$.

Let G be a graph. The functor G.allDPaths() yields a non empty subset of G.allDTrails() and is defined by:

(Def. 38) G.allDPaths() $= \{W : W$ ranges over directed paths of $G\}$.

Let G be a finite graph. One can check that G.allTrails() is finite.

Let G be a graph and let X be a non empty subset of G.allWalks(). We see that the element of X is a walk of G.

Let G be a graph and let X be a non empty subset of G.allTrails(). We see that the element of X is a trail of G.

Let G be a graph and let X be a non empty subset of G.allPaths(). We see that the element of X is a path of G.

Let G be a graph and let X be a non empty subset of G.allDWalks(). We see that the element of X is a dwalk of G.

Let G be a graph and let X be a non empty subset of G.allDTrails(). We see that the element of X is a dtrail of G.

Let G be a graph and let X be a non empty subset of G.allDPaths(). We see that the element of X is a dpath of G.

3. Walk Theorems

For simplicity, we adopt the following rules: G, G_1, G_2 are graphs, W, W_1, W_2 are walks of G, e, x, y, z are sets, v is a vertex of G, and n, m are natural numbers.

We now state a number of propositions:

(8)[3] For every odd natural number n such that $n \leq \operatorname{len} W$ holds $W(n) \in$ the vertices of G.

(9) For every even natural number n such that $n \in \operatorname{dom} W$ holds $W(n) \in$ the edges of G.

(10) Let n be an even natural number. Suppose $n \in \operatorname{dom} W$. Then there exists an odd natural number n_1 such that $n_1 = n - 1$ and $n - 1 \in \operatorname{dom} W$ and $n + 1 \in \operatorname{dom} W$ and $W(n)$ joins $W(n_1)$ and $W(n+1)$ in G.

(11) For every odd natural number n such that $n < \operatorname{len} W$ holds $W(n+1) \in (W.\mathrm{vertexAt}(n)).\mathrm{edgesInOut}()$.

(12) For every odd natural number n such that $1 < n$ and $n \leq \operatorname{len} W$ holds $W(n-1) \in (W.\mathrm{vertexAt}(n)).\mathrm{edgesInOut}()$.

(13) For every odd natural number n such that $n < \operatorname{len} W$ holds $n \in \operatorname{dom} W$ and $n + 1 \in \operatorname{dom} W$ and $n + 2 \in \operatorname{dom} W$.

(14) $\operatorname{len}(G.\mathrm{walkOf}(v)) = 1$ and $(G.\mathrm{walkOf}(v))(1) = v$ and $(G.\mathrm{walkOf}(v)).\mathrm{first}() = v$ and $(G.\mathrm{walkOf}(v)).\mathrm{last}() = v$ and $G.\mathrm{walkOf}(v)$ is walk from v to v.

(15) If e joins x and y in G, then $\operatorname{len}(G.\mathrm{walkOf}(x, e, y)) = 3$.

(16) If e joins x and y in G, then $(G.\mathrm{walkOf}(x, e, y)).\mathrm{first}() = x$ and $(G.\mathrm{walkOf}(x, e, y)).\mathrm{last}() = y$ and $G.\mathrm{walkOf}(x, e, y)$ is walk from x to y.

(17) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds $W_1.\mathrm{first}() = W_2.\mathrm{first}()$ and $W_1.\mathrm{last}() = W_2.\mathrm{last}()$.

(18) W is walk from x to y iff $W(1) = x$ and $W(\operatorname{len} W) = y$.

(19) If W is walk from x to y, then x is a vertex of G and y is a vertex of G.

(20) Let W_1 be a walk of G_1 and W_2 be a walk of G_2. If $W_1 = W_2$, then W_1 is walk from x to y iff W_2 is walk from x to y.

(21) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ and for every natural number n holds $W_1.\mathrm{vertexAt}(n) = W_2.\mathrm{vertexAt}(n)$.

(22) $\operatorname{len} W = \operatorname{len}(W.\mathrm{reverse}())$ and $\operatorname{dom} W = \operatorname{dom}(W.\mathrm{reverse}())$ and $\operatorname{rng} W = \operatorname{rng}(W.\mathrm{reverse}())$.

(23) $W.\mathrm{first}() = W.\mathrm{reverse}().\mathrm{last}()$ and $W.\mathrm{last}() = W.\mathrm{reverse}().\mathrm{first}()$.

(24) W is walk from x to y iff $W.\mathrm{reverse}()$ is walk from y to x.

[3] The proposition (7) has been removed.

(25) If $n \in \mathrm{dom}\, W$, then $W(n) = W.\mathrm{reverse}()((\mathrm{len}\, W - n) + 1)$ and $(\mathrm{len}\, W - n) + 1 \in \mathrm{dom}(W.\mathrm{reverse}())$.

(26) If $n \in \mathrm{dom}(W.\mathrm{reverse}())$, then $W.\mathrm{reverse}()(n) = W((\mathrm{len}\, W - n) + 1)$ and $(\mathrm{len}\, W - n) + 1 \in \mathrm{dom}\, W$.

(27) $W.\mathrm{reverse}().\mathrm{reverse}() = W$.

(28) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds $W_1.\mathrm{reverse}() = W_2.\mathrm{reverse}()$.

(29) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $\mathrm{len}(W_1.\mathrm{append}(W_2)) + 1 = \mathrm{len}\, W_1 + \mathrm{len}\, W_2$.

(30) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $\mathrm{len}\, W_1 \leq \mathrm{len}(W_1.\mathrm{append}(W_2))$ and $\mathrm{len}\, W_2 \leq \mathrm{len}(W_1.\mathrm{append}(W_2))$.

(31) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $(W_1.\mathrm{append}(W_2)).\mathrm{first}() = W_1.\mathrm{first}()$ and $(W_1.\mathrm{append}(W_2)).\mathrm{last}() = W_2.\mathrm{last}()$ and $W_1.\mathrm{append}(W_2)$ is walk from $W_1.\mathrm{first}()$ to $W_2.\mathrm{last}()$.

(32) If W_1 is walk from x to y and W_2 is walk from y to z, then $W_1.\mathrm{append}(W_2)$ is walk from x to z.

(33) If $n \in \mathrm{dom}\, W_1$, then $(W_1.\mathrm{append}(W_2))(n) = W_1(n)$ and $n \in \mathrm{dom}(W_1.\mathrm{append}(W_2))$.

(34) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then for every natural number n such that $n < \mathrm{len}\, W_2$ holds $(W_1.\mathrm{append}(W_2))(\mathrm{len}\, W_1 + n) = W_2(n+1)$ and $\mathrm{len}\, W_1 + n \in \mathrm{dom}(W_1.\mathrm{append}(W_2))$.

(35) If $n \in \mathrm{dom}(W_1.\mathrm{append}(W_2))$, then $n \in \mathrm{dom}\, W_1$ or there exists a natural number k such that $k < \mathrm{len}\, W_2$ and $n = \mathrm{len}\, W_1 + k$.

(36) For all walks W_3, W_4 of G_1 and for all walks W_5, W_6 of G_2 such that $W_3 = W_5$ and $W_4 = W_6$ holds $W_3.\mathrm{append}(W_4) = W_5.\mathrm{append}(W_6)$.

(37) Let m, n be odd natural numbers. Suppose $m \leq n$ and $n \leq \mathrm{len}\, W$. Then $\mathrm{len}(W.\mathrm{cut}(m,n)) + m = n + 1$ and for every natural number i such that $i < \mathrm{len}(W.\mathrm{cut}(m,n))$ holds $(W.\mathrm{cut}(m,n))(i+1) = W(m+i)$ and $m + i \in \mathrm{dom}\, W$.

(38) Let m, n be odd natural numbers. Suppose $m \leq n$ and $n \leq \mathrm{len}\, W$. Then $(W.\mathrm{cut}(m,n)).\mathrm{first}() = W(m)$ and $(W.\mathrm{cut}(m,n)).\mathrm{last}() = W(n)$ and $W.\mathrm{cut}(m,n)$ is walk from $W(m)$ to $W(n)$.

(39) For all odd natural numbers m, n, o such that $m \leq n$ and $n \leq o$ and $o \leq \mathrm{len}\, W$ holds $(W.\mathrm{cut}(m,n)).\mathrm{append}((W.\mathrm{cut}(n,o))) = W.\mathrm{cut}(m,o)$.

(40) $W.\mathrm{cut}(1, \mathrm{len}\, W) = W$.

(41) For every odd natural number n such that $n < \mathrm{len}\, W$ holds $G.\mathrm{walkOf}(W(n), W(n+1), W(n+2)) = W.\mathrm{cut}(n, n+2)$.

(42) For all odd natural numbers m, n such that $m \leq n$ and $n < \mathrm{len}\, W$ holds $(W.\mathrm{cut}(m,n)).\mathrm{addEdge}(W(n+1)) = W.\mathrm{cut}(m, n+2)$.

(43) For every odd natural number n such that $n \leq \operatorname{len} W$ holds $W.\mathrm{cut}(n, n) = \langle W.\mathrm{vertexAt}(n) \rangle$.

(44) If m is odd and $m \leq n$, then $W.\mathrm{cut}(1, n).\mathrm{cut}(1, m) = W.\mathrm{cut}(1, m)$.

(45) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \operatorname{len} W_1$ and $W_1.\mathrm{last}() = W_2.\mathrm{first}()$ holds $(W_1.\mathrm{append}(W_2)).\mathrm{cut}(m, n) = W_1.\mathrm{cut}(m, n)$.

(46) For every odd natural number m such that $m \leq \operatorname{len} W$ holds $\operatorname{len}(W.\mathrm{cut}(1, m)) = m$.

(47) For every odd natural number m and for every natural number x such that $x \in \operatorname{dom}(W.\mathrm{cut}(1, m))$ and $m \leq \operatorname{len} W$ holds $(W.\mathrm{cut}(1, m))(x) = W(x)$.

(48) Let m, n be odd natural numbers and i be a natural number. If $m \leq n$ and $n \leq \operatorname{len} W$ and $i \in \operatorname{dom}(W.\mathrm{cut}(m, n))$, then $(W.\mathrm{cut}(m, n))(i) = W((m + i) - 1)$ and $(m + i) - 1 \in \operatorname{dom} W$.

(49) For every walk W_1 of G_1 and for every walk W_2 of G_2 and for all natural numbers m, n such that $W_1 = W_2$ holds $W_1.\mathrm{cut}(m, n) = W_2.\mathrm{cut}(m, n)$.

(50) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ holds $\operatorname{len}(W.\mathrm{remove}(m, n)) + n = \operatorname{len} W + m$.

(51) If W is walk from x to y, then $W.\mathrm{remove}(m, n)$ is walk from x to y.

(52) $\operatorname{len}(W.\mathrm{remove}(m, n)) \leq \operatorname{len} W$.

(53) $W.\mathrm{remove}(m, m) = W$.

(54) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ holds $(W.\mathrm{cut}(1, m)).\mathrm{last}() = (W.\mathrm{cut}(n, \operatorname{len} W)).\mathrm{first}()$.

(55) Let m, n be odd natural numbers. Suppose $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$. Let x be a natural number. If $x \in \operatorname{Seg} m$, then $(W.\mathrm{remove}(m, n))(x) = W(x)$.

(56) Let m, n be odd natural numbers. Suppose $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$. Let x be a natural number. Suppose $m \leq x$ and $x \leq \operatorname{len}(W.\mathrm{remove}(m, n))$. Then $(W.\mathrm{remove}(m, n))(x) = W((x - m) + n)$ and $(x - m) + n$ is a natural number and $(x - m) + n \leq \operatorname{len} W$.

(57) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ holds $\operatorname{len}(W.\mathrm{remove}(m, n)) = (\operatorname{len} W + m) - n$.

(58) For every natural number m such that $W(m) = W.\mathrm{last}()$ holds $W.\mathrm{remove}(m, \operatorname{len} W) = W.\mathrm{cut}(1, m)$.

(59) For every natural number m such that $W.\mathrm{first}() = W(m)$ holds $W.\mathrm{remove}(1, m) = W.\mathrm{cut}(m, \operatorname{len} W)$.

(60) $(W.\mathrm{remove}(m, n)).\mathrm{first}() = W.\mathrm{first}()$ and $(W.\mathrm{remove}(m, n)).\mathrm{last}() = W.\mathrm{last}()$.

(61) Let m, n be odd natural numbers and x be a natural number. Suppose $m \leq n$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ and $x \in \operatorname{dom}(W.\mathrm{remove}(m, n))$.

Then $x \in \operatorname{Seg} m$ or $m \leq x$ and $x \leq \operatorname{len}(W.\mathrm{remove}(m, n))$.

(62) For every walk W_1 of G_1 and for every walk W_2 of G_2 and for all natural numbers m, n such that $W_1 = W_2$ holds $W_1.\mathrm{remove}(m, n) = W_2.\mathrm{remove}(m, n)$.

(63) If e joins $W.\mathrm{last}()$ and x in G, then $W.\mathrm{addEdge}(e) = W \frown \langle e, x \rangle$.

(64) If e joins $W.\mathrm{last}()$ and x in G, then $(W.\mathrm{addEdge}(e)).\mathrm{first}() = W.\mathrm{first}()$ and $(W.\mathrm{addEdge}(e)).\mathrm{last}() = x$ and $W.\mathrm{addEdge}(e)$ is walk from $W.\mathrm{first}()$ to x.

(65) If e joins $W.\mathrm{last}()$ and x in G, then $\operatorname{len}(W.\mathrm{addEdge}(e)) = \operatorname{len} W + 2$.

(66) Suppose e joins $W.\mathrm{last}()$ and x in G. Then $(W.\mathrm{addEdge}(e))(\operatorname{len} W + 1) = e$ and $(W.\mathrm{addEdge}(e))(\operatorname{len} W + 2) = x$ and for every natural number n such that $n \in \operatorname{dom} W$ holds $(W.\mathrm{addEdge}(e))(n) = W(n)$.

(67) If W is walk from x to y and e joins y and z in G, then $W.\mathrm{addEdge}(e)$ is walk from x to z.

(68) $1 \leq \operatorname{len}(W.\mathrm{vertexSeq}())$.

(69) For every odd natural number n such that $n \leq \operatorname{len} W$ holds $2 \cdot ((n+1) \div 2) - 1 = n$ and $1 \leq (n+1) \div 2$ and $(n+1) \div 2 \leq \operatorname{len}(W.\mathrm{vertexSeq}())$.

(70) $(G.\mathrm{walkOf}(v)).\mathrm{vertexSeq}() = \langle v \rangle$.

(71) If e joins x and y in G, then $(G.\mathrm{walkOf}(x, e, y)).\mathrm{vertexSeq}() = \langle x, y \rangle$.

(72) $W.\mathrm{first}() = W.\mathrm{vertexSeq}()(1)$ and $W.\mathrm{last}() = W.\mathrm{vertexSeq}()(\operatorname{len}(W.\mathrm{vertexSeq}()))$.

(73) For every odd natural number n such that $n \leq \operatorname{len} W$ holds $W.\mathrm{vertexAt}(n) = W.\mathrm{vertexSeq}()((n+1) \div 2)$.

(74) $n \in \operatorname{dom}(W.\mathrm{vertexSeq}())$ iff $2 \cdot n - 1 \in \operatorname{dom} W$.

(75) $(W.\mathrm{cut}(1, n)).\mathrm{vertexSeq}() \subseteq W.\mathrm{vertexSeq}()$.

(76) If e joins $W.\mathrm{last}()$ and x in G, then $(W.\mathrm{addEdge}(e)).\mathrm{vertexSeq}() = W.\mathrm{vertexSeq}() \frown \langle x \rangle$.

(77) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds $W_1.\mathrm{vertexSeq}() = W_2.\mathrm{vertexSeq}()$.

(78) For every even natural number n such that $1 \leq n$ and $n \leq \operatorname{len} W$ holds $n \div 2 \in \operatorname{dom}(W.\mathrm{edgeSeq}())$ and $W(n) = W.\mathrm{edgeSeq}()(n \div 2)$.

(79) $n \in \operatorname{dom}(W.\mathrm{edgeSeq}())$ iff $2 \cdot n \in \operatorname{dom} W$.

(80) For every natural number n such that $n \in \operatorname{dom}(W.\mathrm{edgeSeq}())$ holds $W.\mathrm{edgeSeq}()(n) \in$ the edges of G.

(81) There exists an even natural number l_1 such that $l_1 = \operatorname{len} W - 1$ and $\operatorname{len}(W.\mathrm{edgeSeq}()) = l_1 \div 2$.

(82) $(W.\mathrm{cut}(1, n)).\mathrm{edgeSeq}() \subseteq W.\mathrm{edgeSeq}()$.

(83) If e joins $W.\mathrm{last}()$ and x in G, then $(W.\mathrm{addEdge}(e)).\mathrm{edgeSeq}() = W.\mathrm{edgeSeq}() \frown \langle e \rangle$.

(84) e joins x and y in G iff $(G.\mathrm{walkOf}(x, e, y)).\mathrm{edgeSeq}() = \langle e \rangle$.

(85) $W.\mathrm{reverse}().\mathrm{edgeSeq}() = \mathrm{Rev}(W.\mathrm{edgeSeq}())$.

(86) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $(W_1.\mathrm{append}(W_2)).\mathrm{edgeSeq}() = W_1.\mathrm{edgeSeq}() \frown W_2.\mathrm{edgeSeq}()$.

(87) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds $W_1.\mathrm{edgeSeq}() = W_2.\mathrm{edgeSeq}()$.

(88) $x \in W.\mathrm{vertices}()$ iff there exists an odd natural number n such that $n \leq \mathrm{len}\, W$ and $W(n) = x$.

(89) $W.\mathrm{first}() \in W.\mathrm{vertices}()$ and $W.\mathrm{last}() \in W.\mathrm{vertices}()$.

(90) For every odd natural number n such that $n \leq \mathrm{len}\, W$ holds $W.\mathrm{vertexAt}(n) \in W.\mathrm{vertices}()$.

(91) $(G.\mathrm{walkOf}(v)).\mathrm{vertices}() = \{v\}$.

(92) If e joins x and y in G, then $(G.\mathrm{walkOf}(x, e, y)).\mathrm{vertices}() = \{x, y\}$.

(93) $W.\mathrm{vertices}() = W.\mathrm{reverse}().\mathrm{vertices}()$.

(94) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $(W_1.\mathrm{append}(W_2)).\mathrm{vertices}() = W_1.\mathrm{vertices}() \cup W_2.\mathrm{vertices}()$.

(95) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \mathrm{len}\, W$ holds $(W.\mathrm{cut}(m, n)).\mathrm{vertices}() \subseteq W.\mathrm{vertices}()$.

(96) If e joins $W.\mathrm{last}()$ and x in G, then $(W.\mathrm{addEdge}(e)).\mathrm{vertices}() = W.\mathrm{vertices}() \cup \{x\}$.

(97) Let G be a finite graph, W be a walk of G, and e, x be sets. If e joins $W.\mathrm{last}()$ and x in G and $x \notin W.\mathrm{vertices}()$, then $\mathrm{card}((W.\mathrm{addEdge}(e)).\mathrm{vertices}()) = \mathrm{card}(W.\mathrm{vertices}()) + 1$.

(98) If $x \in W.\mathrm{vertices}()$ and $y \in W.\mathrm{vertices}()$, then there exists a walk of G which is walk from x to y.

(99) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds $W_1.\mathrm{vertices}() = W_2.\mathrm{vertices}()$.

(100) $e \in W.\mathrm{edges}()$ iff there exists an even natural number n such that $1 \leq n$ and $n \leq \mathrm{len}\, W$ and $W(n) = e$.

(101) $e \in W.\mathrm{edges}()$ iff there exists an odd natural number n such that $n < \mathrm{len}\, W$ and $W(n+1) = e$.

(102) $\mathrm{rng}\, W = W.\mathrm{vertices}() \cup W.\mathrm{edges}()$.

(103) If $W_1.\mathrm{last}() = W_2.\mathrm{first}()$, then $(W_1.\mathrm{append}(W_2)).\mathrm{edges}() = W_1.\mathrm{edges}() \cup W_2.\mathrm{edges}()$.

(104) Suppose $e \in W.\mathrm{edges}()$. Then there exist vertices v_2, v_3 of G and there exists an odd natural number n such that $n + 2 \leq \mathrm{len}\, W$ and $v_2 = W(n)$ and $e = W(n+1)$ and $v_3 = W(n+2)$ and e joins v_2 and v_3 in G.

(105) $e \in W.\mathrm{edges}()$ iff there exists a natural number n such that $n \in \mathrm{dom}(W.\mathrm{edgeSeq}())$ and $W.\mathrm{edgeSeq}()(n) = e$.

(106) If $e \in W$.edges() and e joins x and y in G, then $x \in W$.vertices() and $y \in W$.vertices().

(107) (W.cut(m, n)).edges() $\subseteq W$.edges().

(108) W.edges() $= W$.reverse().edges().

(109) e joins x and y in G iff (G.walkOf(x, e, y)).edges() $= \{e\}$.

(110) W.edges() $\subseteq G$.edgesBetween(W.vertices()).

(111) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds W_1.edges() $= W_2$.edges().

(112) If e joins W.last() and x in G, then (W.addEdge(e)).edges() $=$ W.edges() $\cup \{e\}$.

(113) $\operatorname{len} W = 2 \cdot W$.length() $+ 1$.

(114) $\operatorname{len} W_1 = \operatorname{len} W_2$ iff W_1.length() $= W_2$.length().

(115) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ holds W_1.length() $= W_2$.length().

(116) For every odd natural number n such that $n \leq \operatorname{len} W$ holds W.find$(W(n)) \leq n$ and W.rfind$(W(n)) \geq n$.

(117) For every walk W_1 of G_1 and for every walk W_2 of G_2 and for every set v such that $W_1 = W_2$ holds W_1.find$(v) = W_2$.find(v) and W_1.rfind$(v) = W_2$.rfind(v).

(118) For every odd natural number n such that $n \leq \operatorname{len} W$ holds W.find$(n) \leq n$ and W.rfind$(n) \geq n$.

(119) W is closed iff $W(1) = W(\operatorname{len} W)$.

(120) W is closed iff there exists a set x such that W is walk from x to x.

(121) W is closed iff W.reverse() is closed.

(122) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ and W_1 is closed holds W_2 is closed.

(123) W is directed if and only if for every odd natural number n such that $n < \operatorname{len} W$ holds $W(n+1)$ joins $W(n)$ to $W(n+2)$ in G.

(124) Suppose W is directed and walk from x to y and e joins y to z in G. Then W.addEdge(e) is directed and W.addEdge(e) is walk from x to z.

(125) For every dwalk W of G and for all natural numbers m, n holds W.cut(m, n) is directed.

(126) W is non trivial iff $3 \leq \operatorname{len} W$.

(127) W is non trivial iff $\operatorname{len} W \neq 1$.

(128) If W.first() $\neq W$.last(), then W is non trivial.

(129) W is trivial iff there exists a vertex v of G such that $W = G$.walkOf(v).

(130) W is trivial iff W.reverse() is trivial.

(131) If W_2 is trivial, then W_1.append$(W_2) = W_1$.

(132) For all odd natural numbers m, n such that $m \leq n$ and $n \leq \operatorname{len} W$ holds W.cut(m, n) is trivial iff $m = n$.

(133) If e joins W.last() and x in G, then W.addEdge(e) is non trivial.

(134) If W is non trivial, then there exists an odd natural number l_2 such that $l_2 = \operatorname{len} W - 2$ and $(W.\text{cut}(1, l_2))$.addEdge$(W(l_2 + 1)) = W$.

(135) If W_2 is non trivial and W_2.edges() $\subseteq W_1$.edges(), then W_2.vertices() $\subseteq W_1$.vertices().

(136) If W is non trivial, then for every vertex v of G such that $v \in W$.vertices() holds v is not isolated.

(137) W is trivial iff W.edges() $= \emptyset$.

(138) For every walk W_1 of G_1 and for every walk W_2 of G_2 such that $W_1 = W_2$ and W_1 is trivial holds W_2 is trivial.

(139) W is trail-like iff for all even natural numbers m, n such that $1 \leq m$ and $m < n$ and $n \leq \operatorname{len} W$ holds $W(m) \neq W(n)$.

(140) If $\operatorname{len} W \leq 3$, then W is trail-like.

(141) W is trail-like iff W.reverse() is trail-like.

(142) For every trail W of G and for all natural numbers m, n holds W.cut(m, n) is trail-like.

(143) For every trail W of G and for every set e such that $e \in W$.last().edgesInOut() and $e \notin W$.edges() holds W.addEdge(e) is trail-like.

(144) For every trail W of G and for every vertex v of G such that $v \in W$.vertices() and v is endvertex holds $v = W$.first() or $v = W$.last().

(145) For every finite graph G and for every trail W of G holds len$(W$.edgeSeq()$) \leq G$.size().

(146) If $\operatorname{len} W \leq 3$, then W is path-like.

(147) If for all odd natural numbers m, n such that $m \leq \operatorname{len} W$ and $n \leq \operatorname{len} W$ and $W(m) = W(n)$ holds $m = n$, then W is path-like.

(148) Let W be a path of G. Suppose W is open. Let m, n be odd natural numbers. If $m < n$ and $n \leq \operatorname{len} W$, then $W(m) \neq W(n)$.

(149) W is path-like iff W.reverse() is path-like.

(150) For every path W of G and for all natural numbers m, n holds W.cut(m, n) is path-like.

(151) Let W be a path of G and e, v be sets. Suppose that
 (i) e joins W.last() and v in G,
 (ii) $e \notin W$.edges(),
 (iii) W is trivial or open, and
 (iv) for every odd natural number n such that $1 < n$ and $n \leq \operatorname{len} W$ holds $W(n) \neq v$.

Then W.addEdge(e) is path-like.

(152) Let W be a path of G and e, v be sets. Suppose e joins W.last() and v in G and $v \notin W$.vertices() and W is trivial or open. Then W.addEdge(e) is path-like.

(153) If for every odd natural number n such that $n \leq \operatorname{len} W$ holds W.find($W(n)$) $=$ W.rfind($W(n)$), then W is path-like.

(154) If for every odd natural number n such that $n \leq \operatorname{len} W$ holds W.rfind(n) $= n$, then W is path-like.

(155) For every finite graph G and for every path W of G holds len(W.vertexSeq()) $\leq G$.order() $+ 1$.

(156) Let G be a graph, W be a vertex-distinct walk of G, and e, v be sets. If e joins W.last() and v in G and $v \notin W$.vertices(), then W.addEdge(e) is vertex-distinct.

(157) If e joins x and x in G, then G.walkOf(x, e, x) is cycle-like.

(158) Suppose e joins x and y in G and $e \in W_1$.edges() and W_1 is cycle-like. Then there exists a walk W_2 of G such that W_2 is walk from x to y and $e \notin W_2$.edges().

(159) W is a subwalk of W.

(160) For every walk W_1 of G and for every subwalk W_2 of W_1 holds every subwalk of W_2 is a subwalk of W_1.

(161) If W_1 is a subwalk of W_2, then W_1 is walk from x to y iff W_2 is walk from x to y.

(162) If W_1 is a subwalk of W_2, then W_1.first() $=$ W_2.first() and W_1.last() $=$ W_2.last().

(163) If W_1 is a subwalk of W_2, then $\operatorname{len} W_1 \leq \operatorname{len} W_2$.

(164) If W_1 is a subwalk of W_2, then W_1.edges() $\subseteq$ W_2.edges() and W_1.vertices() $\subseteq$ W_2.vertices().

(165) Suppose W_1 is a subwalk of W_2. Let m be an odd natural number. Suppose $m \leq \operatorname{len} W_1$. Then there exists an odd natural number n such that $m \leq n$ and $n \leq \operatorname{len} W_2$ and $W_1(m) = W_2(n)$.

(166) Suppose W_1 is a subwalk of W_2. Let m be an even natural number. Suppose $1 \leq m$ and $m \leq \operatorname{len} W_1$. Then there exists an even natural number n such that $m \leq n$ and $n \leq \operatorname{len} W_2$ and $W_1(m) = W_2(n)$.

(167) For every trail W_1 of G such that W_1 is non trivial holds there exists a path of W_1 which is non trivial.

(168) For every graph G_1 and for every subgraph G_2 of G_1 holds every walk of G_2 is a walk of G_1.

(169) Let G_1 be a graph, G_2 be a subgraph of G_1, and W be a walk of G_1. If W is trivial and W.first() $\in$ the vertices of G_2, then W is a walk of G_2.

(170) Let G_1 be a graph, G_2 be a subgraph of G_1, and W be a walk of G_1. If W is non trivial and W.edges() $\subseteq$ the edges of G_2, then W is a walk of G_2.

(171) Let G_1 be a graph, G_2 be a subgraph of G_1, and W be a walk of G_1. Suppose W.vertices() $\subseteq$ the vertices of G_2 and W.edges() $\subseteq$ the edges of G_2. Then W is a walk of G_2.

(172) Let G_1 be a non trivial graph, W be a walk of G_1, v be a vertex of G_1, and G_2 be a subgraph of G_1 with vertex v removed. If $v \notin W$.vertices(), then W is a walk of G_2.

(173) Let G_1 be a graph, W be a walk of G_1, e be a set, and G_2 be a subgraph of G_1 with edge e removed. If $e \notin W$.edges(), then W is a walk of G_2.

(174) Let G_1 be a graph, G_2 be a subgraph of G_1, and x, y, e be sets. If e joins x and y in G_2, then G_1.walkOf$(x, e, y) = G_2$.walkOf(x, e, y).

(175) Let G_1 be a graph, G_2 be a subgraph of G_1, W_1 be a walk of G_1, W_2 be a walk of G_2, and e be a set. If $W_1 = W_2$ and $e \in W_2$.last().edgesInOut(), then W_1.addEdge$(e) = W_2$.addEdge(e).

(176) Let G_1 be a graph, G_2 be a subgraph of G_1, and W be a walk of G_2. Then
  (i) if W is closed, then W is a closed walk of G_1,
  (ii) if W is directed, then W is a directed walk of G_1,
  (iii) if W is trivial, then W is a trivial walk of G_1,
  (iv) if W is trail-like, then W is a trail-like walk of G_1,
  (v) if W is path-like, then W is a path-like walk of G_1, and
  (vi) if W is vertex-distinct, then W is a vertex-distinct walk of G_1.

(177) Let G_1 be a graph, G_2 be a subgraph of G_1, W_1 be a walk of G_1, and W_2 be a walk of G_2 such that $W_1 = W_2$. Then
  (i) W_1 is closed iff W_2 is closed,
  (ii) W_1 is directed iff W_2 is directed,
  (iii) W_1 is trivial iff W_2 is trivial,
  (iv) W_1 is trail-like iff W_2 is trail-like,
  (v) W_1 is path-like iff W_2 is path-like, and
  (vi) W_1 is vertex-distinct iff W_2 is vertex-distinct.

(178) If $G_1 =_G G_2$ and x is a vertex sequence of G_1, then x is a vertex sequence of G_2.

(179) If $G_1 =_G G_2$ and x is a edge sequence of G_1, then x is a edge sequence of G_2.

(180) If $G_1 =_G G_2$ and x is a walk of G_1, then x is a walk of G_2.

(181) If $G_1 =_G G_2$, then G_1.walkOf$(x, e, y) = G_2$.walkOf(x, e, y).

(182) Let W_1 be a walk of G_1 and W_2 be a walk of G_2 such that $G_1 =_G G_2$ and $W_1 = W_2$. Then

(i) W_1 is closed iff W_2 is closed,
(ii) W_1 is directed iff W_2 is directed,
(iii) W_1 is trivial iff W_2 is trivial,
(iv) W_1 is trail-like iff W_2 is trail-like,
(v) W_1 is path-like iff W_2 is path-like, and
(vi) W_1 is vertex-distinct iff W_2 is vertex-distinct.

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. The ordinal numbers. *Formalized Mathematics*, 1(**1**):91–96, 1990.
[4] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.
[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. Some properties of restrictions of finite sequences. *Formalized Mathematics*, 5(**2**):241–245, 1996.
[8] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[9] Gilbert Lee and Piotr Rudnicki. Alternative graph structures. *Formalized Mathematics*, 13(**2**):235–252, 2005.
[10] Yatsuka Nakamura and Piotr Rudnicki. Vertex sequences induced by chains. *Formalized Mathematics*, 5(**3**):297–304, 1996.
[11] Piotr Rudnicki and Andrzej Trybulec. Abian's fixed point theorem. *Formalized Mathematics*, 6(**3**):335–338, 1997.
[12] Andrzej Trybulec. Subsets of complex numbers. *To appear in Formalized Mathematics.*
[13] Andrzej Trybulec. Domains and their Cartesian products. *Formalized Mathematics*, 1(**1**):115–122, 1990.
[14] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[15] Andrzej Trybulec. Tuples, projections and Cartesian products. *Formalized Mathematics*, 1(**1**):97–105, 1990.
[16] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[17] Josef Urban. Basic facts about inaccessible and measurable cardinals. *Formalized Mathematics*, 9(**2**):323–329, 2001.
[18] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.
[19] Edmund Woronowicz. Relations defined on sets. *Formalized Mathematics*, 1(**1**):181–186, 1990.

Received February 22, 2005