FORMALIZED MATHEMATICS
Volume 13, Number 2, Pages 271–277
University of Białystok, 2005

Trees and Graph Components[1]

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

Summary. In the graph framework of [11] we define connected and acyclic graphs, components of a graph, and define the notion of cut-vertex (articulation point).

MML identifier: GLIB_002, version: 7.5.01 4.39.921

The articles [15], [8], [14], [17], [12], [18], [6], [1], [16], [7], [3], [4], [5], [9], [2], [11], [10], and [13] provide the terminology and notation for this paper.

1. Preliminaries

Let X be a finite set. Observe that 2^X is finite.

The following proposition is true

(1) For every finite set X such that $1 < \operatorname{card} X$ there exist sets x_1, x_2 such that $x_1 \in X$ and $x_2 \in X$ and $x_1 \neq x_2$.

2. Definitions

Let G be a graph. We say that G is connected if and only if:

(Def. 1) For all vertices u, v of G holds there exists a walk of G which is walk from u to v.

Let G be a graph. We say that G is acyclic if and only if:

(Def. 2) There exists no walk of G which is cycle-like.

Let G be a graph. We say that G is tree-like if and only if:

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

ISSN 1426–2630

(Def. 3) G is acyclic and connected.

One can verify that every graph which is trivial is also connected.

Let us note that every graph which is trivial and loopless is also tree-like.

Let us note that every graph which is acyclic is also simple.

Let us observe that every graph which is tree-like is also acyclic and connected.

Let us observe that every graph which is acyclic and connected is also tree-like.

Let G be a graph and let v be a vertex of G. Observe that every subgraph of G induced by $\{v\}$ and $\emptyset$ is tree-like.

Let G be a graph and let v be a set. We say that G is dtree rooted at v if and only if:

(Def. 4) G is tree-like and for every vertex x of G holds there exists a dwalk of G which is walk from v to x.

Let us observe that there exists a graph which is trivial, finite, and tree-like and there exists a graph which is non trivial, finite, and tree-like.

Let G be a graph. Note that there exists a subgraph of G which is trivial, finite, and tree-like.

Let G be an acyclic graph. Observe that every subgraph of G is acyclic.

Let G be a graph and let v be a vertex of G. The functor G.reachableFrom(v) yields a non empty subset of the vertices of G and is defined as follows:

(Def. 5) For every set x holds $x \in G$.reachableFrom(v) iff there exists a walk of G which is walk from v to x.

Let G be a graph and let v be a vertex of G. The functor G.reachableDFrom(v) yielding a non empty subset of the vertices of G is defined by:

(Def. 6) For every set x holds $x \in G$.reachableDFrom(v) iff there exists a dwalk of G which is walk from v to x.

Let G_1 be a graph and let G_2 be a subgraph of G_1. We say that G_2 is component-like if and only if:

(Def. 7) G_2 is connected and it is not true that there exists a connected subgraph G_3 of G_1 such that $G_2 \subset G_3$.

Let G be a graph. Note that every subgraph of G which is component-like is also connected.

Let G be a graph and let v be a vertex of G. Note that every subgraph of G induced by G.reachableFrom(v) is component-like.

Let G be a graph. Observe that there exists a subgraph of G which is component-like.

Let G be a graph. A component of G is a component-like subgraph of G.

Let G be a graph. The functor G.componentSet() yielding a non empty family of subsets of the vertices of G is defined as follows:

(Def. 8) For every set x holds $x \in G$.componentSet() iff there exists a vertex v of G such that $x = G$.reachableFrom(v).

Let G be a graph and let X be an element of G.componentSet(). Observe that every subgraph of G induced by X is component-like.

Let G be a graph. The functor G.numComponents() yielding a cardinal number is defined by:

(Def. 9) $G.\mathrm{numComponents}() = \overline{\overline{G.\mathrm{componentSet}()}}$.

Let G be a finite graph. Then G.numComponents() is a non empty natural number.

Let G be a graph and let v be a vertex of G. We say that v is cut-vertex if and only if:

(Def. 10) For every subgraph G_2 of G with vertex v removed holds $G.\mathrm{numComponents}() < G_2.\mathrm{numComponents}()$.

Let G be a finite graph and let v be a vertex of G. Let us observe that v is cut-vertex if and only if:

(Def. 11) For every subgraph G_2 of G with vertex v removed holds $G.\mathrm{numComponents}() < G_2.\mathrm{numComponents}()$.

Let G be a non trivial finite connected graph. Observe that there exists a vertex of G which is non cut-vertex.

Let G be a non trivial finite tree-like graph. One can check that there exists a vertex of G which is endvertex.

Let G be a non trivial finite tree-like graph and let v be an endvertex vertex of G. Observe that every subgraph of G with vertex v removed is tree-like.

Let G_4 be a graph sequence. We say that G_4 is connected if and only if:

(Def. 12) For every natural number n holds $G_4 \rightarrow n$ is connected.

We say that G_4 is acyclic if and only if:

(Def. 13) For every natural number n holds $G_4 \rightarrow n$ is acyclic.

We say that G_4 is tree-like if and only if:

(Def. 14) For every natural number n holds $G_4 \rightarrow n$ is tree-like.

One can check the following observations:

* every graph sequence which is trivial is also connected,
* every graph sequence which is trivial and loopless is also tree-like,
* every graph sequence which is acyclic is also simple,
* every graph sequence which is tree-like is also acyclic and connected, and
* every graph sequence which is acyclic and connected is also tree-like.

Let us note that there exists a graph sequence which is halting, finite, and tree-like.

Let G_4 be a connected graph sequence and let n be a natural number. Note that $G_4 \rightarrow n$ is connected.

Let G_4 be an acyclic graph sequence and let n be a natural number. Observe that $G_4 \rightarrow n$ is acyclic.

Let G_4 be a tree-like graph sequence and let n be a natural number. Note that $G_4 \rightarrow n$ is tree-like.

3. Theorems

For simplicity, we use the following convention: G, G_1, G_2 are graphs, e, x, y are sets, v, v_1, v_2 are vertices of G, and W is a walk of G.

We now state a number of propositions:

(2) For every non trivial connected graph G and for every vertex v of G holds v is not isolated.

(3) Let G_1 be a non trivial graph, v be a vertex of G_1, and G_2 be a subgraph of G_1 with vertex v removed. Suppose G_2 is connected and there exists a set e such that $e \in v$.edgesInOut() and e does not join v and v in G_1. Then G_1 is connected.

(4) Let G_1 be a non trivial connected graph, v be a vertex of G_1, and G_2 be a subgraph of G_1 with vertex v removed. If v is endvertex, then G_2 is connected.

(5) Let G_1 be a connected 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 W is cycle-like and $e \in W$.edges(), then G_2 is connected.

(6) If there exists a vertex v_1 of G such that for every vertex v_2 of G holds there exists a walk of G which is walk from v_1 to v_2, then G is connected.

(7) Every trivial graph is connected.

(8) If $G_1 =_G G_2$ and G_1 is connected, then G_2 is connected.

(9) $v \in G$.reachableFrom(v).

(10) If $x \in G$.reachableFrom(v_1) and e joins x and y in G, then $y \in G$.reachableFrom(v_1).

(11) G.edgesBetween(G.reachableFrom(v)) = G.edgesInOut(G.reachableFrom(v)).

(12) If $v_1 \in G$.reachableFrom(v_2), then G.reachableFrom(v_1) = G.reachableFrom(v_2).

(13) If $v \in W$.vertices(), then W.vertices() $\subseteq G$.reachableFrom(v).

(14) Let G_1 be a graph, G_2 be a subgraph of G_1, v_1 be a vertex of G_1, and v_2 be a vertex of G_2. If $v_1 = v_2$, then G_2.reachableFrom(v_2) $\subseteq$ G_1.reachableFrom(v_1).

(15) If there exists a vertex v of G such that G.reachableFrom(v) = the vertices of G, then G is connected.

(16) If G is connected, then for every vertex v of G holds G.reachableFrom(v) = the vertices of G.

(17) For every vertex v_1 of G_1 and for every vertex v_2 of G_2 such that $G_1 =_G G_2$ and $v_1 = v_2$ holds G_1.reachableFrom(v_1) = G_2.reachableFrom(v_2).

(18) $v \in G$.reachableDFrom(v).

(19) If $x \in G$.reachableDFrom(v_1) and e joins x to y in G, then $y \in G$.reachableDFrom(v_1).

(20) G.reachableDFrom(v) $\subseteq G$.reachableFrom(v).

(21) Let G_1 be a graph, G_2 be a subgraph of G_1, v_1 be a vertex of G_1, and v_2 be a vertex of G_2. If $v_1 = v_2$, then G_2.reachableDFrom(v_2) $\subseteq$ G_1.reachableDFrom(v_1).

(22) For every vertex v_1 of G_1 and for every vertex v_2 of G_2 such that $G_1 =_G G_2$ and $v_1 = v_2$ holds G_1.reachableDFrom(v_1) = G_2.reachableDFrom(v_2).

(23) For every graph G_1 and for every connected subgraph G_2 of G_1 such that G_2 is spanning holds G_1 is connected.

(24) $\bigcup$(G.componentSet()) = the vertices of G.

(25) G is connected iff G.componentSet() = {the vertices of G}.

(26) If $G_1 =_G G_2$, then G_1.componentSet() = G_2.componentSet().

(27) If $x \in G$.componentSet(), then x is a non empty subset of the vertices of G.

(28) G is connected iff G.numComponents() = 1.

(29) If $G_1 =_G G_2$, then G_1.numComponents() = G_2.numComponents().

(30) G is a component of G iff G is connected.

(31) For every component C of G holds the edges of C = G.edgesBetween(the vertices of C).

(32) For all components C_1, C_2 of G holds the vertices of C_1 = the vertices of C_2 iff $C_1 =_G C_2$.

(33) Let C be a component of G and v be a vertex of G. Then $v \in$ the vertices of C if and only if the vertices of C = G.reachableFrom(v).

(34) Let C_1, C_2 be components of G and v be a set. If $v \in$ the vertices of C_1 and $v \in$ the vertices of C_2, then $C_1 =_G C_2$.

(35) Let G be a connected graph and v be a vertex of G. Then v is non cut-vertex if and only if for every subgraph G_2 of G with vertex v removed holds G_2.numComponents() $\leq G$.numComponents().

(36) Let G be a connected graph, v be a vertex of G, and G_2 be a subgraph of G with vertex v removed. If v is not cut-vertex, then G_2 is connected.

(37) Let G be a non trivial finite connected graph. Then there exist vertices v_1, v_2 of G such that $v_1 \neq v_2$ and v_1 is not cut-vertex and v_2 is not cut-vertex.

(38) If v is cut-vertex, then G is non trivial.

(39) Let v_1 be a vertex of G_1 and v_2 be a vertex of G_2. If $G_1 =_G G_2$ and $v_1 = v_2$, then if v_1 is cut-vertex, then v_2 is cut-vertex.

(40) For every finite connected graph G holds G.order() $\leq G$.size() $+ 1$.

(41) Every acyclic graph is simple.

(42) Let G be an acyclic graph, W be a path of G, and e be a set. If $e \notin W$.edges() and $e \in W$.last().edgesInOut(), then W.addEdge(e) is path-like.

(43) Let G be a non trivial finite acyclic graph. Suppose the edges of $G \neq \emptyset$. Then there exist vertices v_1, v_2 of G such that $v_1 \neq v_2$ and v_1 is endvertex and v_2 is endvertex and $v_2 \in G$.reachableFrom(v_1).

(44) If $G_1 =_G G_2$ and G_1 is acyclic, then G_2 is acyclic.

(45) Let G be a non trivial finite tree-like graph. Then there exist vertices v_1, v_2 of G such that $v_1 \neq v_2$ and v_1 is endvertex and v_2 is endvertex.

(46) For every finite graph G holds G is tree-like iff G is acyclic and G.order() $= G$.size() $+ 1$.

(47) For every finite graph G holds G is tree-like iff G is connected and G.order() $= G$.size() $+ 1$.

(48) If $G_1 =_G G_2$ and G_1 is tree-like, then G_2 is tree-like.

(49) If G is dtree rooted at x, then x is a vertex of G.

(50) If $G_1 =_G G_2$ and G_1 is dtree rooted at x, then G_2 is dtree rooted at x.

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. Sequences of ordinal numbers. *Formalized Mathematics*, 1(**2**):281–290, 1990.
[5] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 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. Some basic properties of sets. *Formalized Mathematics*, 1(**1**):47–53, 1990.
[9] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[10] Gilbert Lee. Walks in Graphs. *Formalized Mathematics*, 13(**2**):253–269, 2005.
[11] Gilbert Lee and Piotr Rudnicki. Alternative graph structures. *Formalized Mathematics*, 13(**2**):235–252, 2005.
[12] Beata Padlewska. Families of sets. *Formalized Mathematics*, 1(**1**):147–152, 1990.
[13] Piotr Rudnicki and Andrzej Trybulec. Abian's fixed point theorem. *Formalized Mathematics*, 6(**3**):335–338, 1997.
[14] Andrzej Trybulec. Subsets of complex numbers. *To appear in Formalized Mathematics.*
[15] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.

[16] Andrzej Trybulec. Many-sorted sets. *Formalized Mathematics*, 4(**1**):15–22, 1993.
[17] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[18] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received February 0, 2005