# Contents

# Chapter 1

# Problem Set 1

1.) Prove Proposition 6: For any $a, b \in \mathbb{R}$, $\begin{pmatrix} a \vee b = \frac{1}{2}(a + b + |a - b|) \\ a \wedge b = \frac{1}{2}(a + b - |a - b|) \end{pmatrix}$.

Let $a, b \in \mathbb{R}$. Suppose $a \geq b$. Clearly, $a \vee b = a$ and $a \wedge b = b$. Furthermore, $|a - b| = a - b$ since $a - b \geq 0$. Thus,

$$a \vee b = a = \frac{1}{2}(a + b + (a - b)) = \frac{1}{2}(a + b + |a - b|) \tag{1.1}$$

$$a \wedge b = b = \frac{1}{2}(a + b - (a - b)) = \frac{1}{2}(a + b - |a - b|). \tag{1.2}$$

Suppose $a < b$. Clearly, $a \vee b = b$ and $a \wedge b = a$. Furthermore, $|a - b| = -(a - b)$ since $a - b < 0$. Thus,

$$a \vee b = b = \frac{1}{2}(a + b + (-(a - b))) = \frac{1}{2}(a + b + |a - b|) \tag{1.3}$$

$$a \wedge b = a = \frac{1}{2}(a + b - (-(a - b))) = \frac{1}{2}(a + b - |a - b|). \tag{1.4}$$

In either case, $a \vee b = \frac{1}{2}(a + b + |a - b|)$ and $a \wedge b = \frac{1}{2}(a + b - |a - b|)$ holds. $\blacksquare$

2.) Prove Proposition 7: For any $a, b, r \in \mathbb{R}$, $\begin{pmatrix} a \vee b = b \vee a \\ a \wedge b = b \wedge a \\ (a \wedge b \leq r \leq a \vee b) \implies ((|r - a| \leq |a - b|) \wedge (r - b \leq |a - b|)) \end{pmatrix}$.

Let $a, b, r \in \mathbb{R}$. The first two statements immediately follow from applying the commutativity of real numbers and $|a - b| = |-(a - b)| = |b - a|$ to Proposition 6.

Suppose $a \wedge b \leq r \leq a \vee b$. Without loss of generality, let $a \geq b$. Thus,

$$b \leq r \leq a \tag{1.5}$$

$$r - a \leq 0 \tag{1.6}$$

$$b - r \leq 0 \tag{1.7}$$

$$b - a \leq 0 \tag{1.8}$$

From (1.5), $r - a \geq b - a$. This along with (1.6) and (1.8) implies $|r - a| = -(r - a) \leq -(b - a) = |b - a|$.
From (1.5), $r - b \leq a - b$. This along with (1.7) and (1.8) implies $|r - b| = r - b \leq a - b = |a - b|$. $\blacksquare$

# Chapter 2

# Graph Theory

## 2.1 Graphs

### 2.1.1 **Graph operations**

$GraphPower[G^r, r, G] := (V = V(G)) \land (E = \{\{x, y\} \mid d(x, y) \leq r\}) \land (G^r = (V, E))$
$GraphSum[G_1 + G_2, G_1, G_2] := (V = V(G_1) \cup V(G_2)) \land (E = E(G_1) \cup E(G_2) \cup \{\{x, y\} \mid (x \in V(G_1)) \land y \in V(G_2)\}) \land (G_1 + G_2 = (V, E))$

$$GraphCartesian[G_1 \times G_2, G_1, G_2] := \begin{pmatrix} (V = V(G_1) \times V(G_2)) & \land \\ (E = \{((x_1, y_1), (x_2, y_2)) \mid ((x_1 = x_2) \land (\{y_1, y_2\} \in E(G_2))) \lor ((y_1 = y_2) \land (\{x_1, x_2\} \in E(G_1)))\}) \land \\ (G_1 \times G_2 = (V, E)) \end{pmatrix}$$

$$GraphComposition[G_1 \circ G_2, G_1, G_2] := \begin{pmatrix} (V = V(G_1) \times V(G_2)) & \land \\ (E = \{((x_1, y_1), (x_2, y_2)) \mid ((x_1 = x_2) \land (\{y_1, y_2\} \in E(G_2))) \lor (\{x_1, x_2\} \in E(G_1))\}) \land \\ (G_1 \circ G_2 = (V, E)) \end{pmatrix}$$

$$GraphConjunction[G_1 \land G_2, G_1, G_2] := \begin{pmatrix} (V = V(G_1) \times V(G_2)) & \land \\ (E = \{((x_1, y_1), (x_2, y_2)) \mid (\{x_1, x_2\} \in E(G_1)) \land (\{y_1, y_2\} \in E(G_2))\}) \land \\ (G_1 \land G_2 = (V, E)) \end{pmatrix}$$

$$KroneckerProduct[A \otimes B, A, B] := (Matrix[A, m, n]) \land (Matrix[B, p, q]) \land \left(A \otimes B = \begin{bmatrix} a_{1,1}B & \dots & a_{1,n}B \\ \vdots & \ddots & \vdots \\ a_{m,1}B & \dots & a_{m,n}B \end{bmatrix} \in \mathbb{R}^{mp} \times \mathbb{R}^{nq}\right)$$

$AdjacencyKroneckerIdentity := \forall_{G,H}(\mathcal{A}(G \land H) = \mathcal{A}(H) \otimes \mathcal{A}(G))$

(1) TODO: https://archive.siam.org/books/textbooks/OT91sample.pdf, etc.

### 2.1.2 Graphs

$SimpleGraph[(V, E)] := (Set[V]) \land (E \subseteq \{\{a, b\} \in V^{\{2\}} \mid a \neq b\})$
$VertexSet[V((V, E)), (V, E)] := (SimpleGraph[(V, E)]) \land (V((V, E)) = V)$
$EdgeSet[E((V, E)), (V, E)] := (SimpleGraph[(V, E)]) \land (E((V, E)) = E)$
$AdjacentV[\{x, y\}, G] := \{x, y\} \in E(G)$
$Incident[e, x, y, G] := e = \{x, y\} \in E(G)$
$Degree[d(x), x, G] := d(x) = |\{y \in V(G) \mid AdjacentV[\{x, y\}, G]\}|$
$Order[n(G), G] := n(G) = |V(G)|$
$Size[e(G), G] := e(G) = |E(G)|$

$ComplementG[\bar{G}, G] := \bar{G} = (V, V^{\{2\}} \setminus (E \cup \{\{x, x\} \mid x \in V(G)\}))$
$Clique[X, G] := \forall_{x_1, x_2 \in X}(AdjacentV[\{x_1, x_2\}, G])$
$IndependentSet[X, G] := \forall_{x_1, x_2 \in X}(\neg AdjacentV[\{x_1, x_2\}, G])$
$BipartiteG[G] := \exists_{X,Y}((IndependentSet[X, G]) \land (IndependentSet[Y, G]) \land (V(G) = X \dot{\cup} Y))$
$Coloring[\phi, C, G] := (Function[\phi, V(G), C]) \land (\forall_{\{x,y\} \in E(G)}(\phi(x) \neq \phi(y)))$
$ChromaticNumber[\chi(G), G] := \chi(G) = min(\{|C| \mid \exists_{\phi,C}(Coloring[\phi, C, G])\})$

$$kPartiteG[G, k] := \exists_{\mathcal{S}}((|\mathcal{S}| = k) \land (\forall_{S \in \mathcal{S}}(IndependentSet[S, G])) \land (V(G) = \dot{\bigcup_{S \in \mathcal{S}}} (S)))$$

$$PartiteSets[\mathcal{S}, G] := (\forall_{S \in \mathcal{S}}(IndependentSet[S, G])) \land (V(G) = \dot{\bigcup_{S \in \mathcal{S}}} (S))$$

$CompleteBipartiteG[G, X, Y] := (PartiteSets[\{X, Y\}, G]) \land (E(G) = \{\{x, y\} \mid (x \in X) \land (y \in Y)\})$

### 2.1.3 Paths, Cycles, Trails

$PathG[G] := \exists_P((Ordering[P,V(G)]) \wedge (E(G) = \{\{p_i, p_{i+1}\} \mid i \in \mathbb{N}_1^{|P|-1}\}))$
$CycleG[G] := \exists_C((Ordering[C,V(G)]) \wedge (E(G) = \{\{c_i, c_{i+1}\} \mid i \in \mathbb{N}_1^{|C|-1}\} \cup \{c_n, c_1\}))$
$CompleteG[G] := \forall_{x,y \in V(G)}((x \neq y) \implies \{x,y\} \in E(G))$
$TriangleG[G] := (CompleteG[G]) \wedge (n(G) = 3)$

$Subgraph[H,G] := (V(H) \subseteq V(G)) \wedge (E(H) \subseteq E(G))$
$ConnectedV[\{x,y\},G] := \exists H((Subgraph[H,G]) \wedge (PathG[H]) \wedge (\{x,y\} \subseteq V(H)))$
$ConnectedG[G] := \forall_{x,y \in V(G)}(ConnectedV[\{x,y\},G])$

$$AdjacencyMatrix[\mathcal{A}(G),G] := (Matrix[\mathcal{A}(G)], n(G), n(G)) \wedge \left( \mathcal{A}(G)_{i,j} = \begin{cases} 1 & \{v_i, v_j\} \in E(G) \\ 0 & \{v_i, v_j\} \notin E(G) \end{cases} \right)$$

$$IncidenceMatrix[\mathcal{I}(G),G] := (Matrix[\mathcal{A}(G)], n(G), e(G)) \wedge \left( \mathcal{I}(G)_{i,j} = \begin{cases} 1 & v_i \in e_j \\ 0 & v_i \notin e_j \end{cases} \right)$$

$Isomorphism[\phi,G,H] := (Bijection[\phi,V(G),V(H)]) \wedge (\forall_{x,y \in V(G)}((\{x,y\} \in E(G)) \iff (\{\phi(x),\phi(y)\} \in E(H))))$
$Isomorphic[G,H] := \exists_\phi(Isomorphism[\phi,G,H])$

$$IsomorphismEqRel := \forall_{G_1,G_2,G_3} \begin{pmatrix} (G_1 \cong G_1) & \wedge \\ ((G_1 \cong G_2) \implies (G_2 \cong G_1)) & \wedge \\ (((G_1 \cong G_2) \wedge (G_2 \cong G_3)) \implies (G_1 \cong G_3)) & \end{pmatrix}$$

(1) Bijection and composition properties

$IsomorphismClass[\mathcal{G}] := (G \in \mathcal{G}) \wedge (\mathcal{G} = [G]_{\cong})$

$PathN[P_n, n] := (PathG[P_n]) \wedge (n(P_n) = n)$
$CycleN[C_n, n] := (CycleG[C_n]) \wedge (n(C_n) = n)$
$CompleteN[K_n, n] := (CompleteG[K_n]) \wedge (n(K_n) = n)$
$BicliqueRS[K_{r,s}, r, s] := (CompleteBipartiteG[K_{r,s}]) \wedge (PartiteSets[\{R,S\},G]) \wedge (|R| = r) \wedge (|S| = s)$

$SelfComplementary[G] := G \cong \bar{G}$
$Decomposition[\mathcal{D},G] := (\forall_{D \in \mathcal{D}}(Subgraph[D,G])) \wedge (\forall_{e \in E(G)} \exists!_{D \in \mathcal{D}}(e \in E(D)))$

TODO: ADD SPECIAL GRAPHS

$$Girth[girth(G),G] := (CycleLengths[L,G]) \wedge \left( girth(G) = \begin{cases} min(L) & L \neq \emptyset \\ \infty & L = \emptyset \end{cases} \right)$$

$$Circumference[circumference(G),G] := (CycleLengths[L,G]) \wedge \left( circumference(G) = \begin{cases} max(L) & L \neq \emptyset \\ \infty & L = \emptyset \end{cases} \right)$$

$Automorphism[\phi,G] := (Isomorphism[\phi,G,G])$
$VertexTransitive[G] := \forall_{x,y \in V(G)} \exists_\phi((Automorphism[\phi,G]) \wedge (\phi(x) = y))$

$Walk[W,G] := (\forall_{i \in \mathbb{N}_1^{|W|-1}}(\{w_i, w_{i+1}\} \in E(G)))$
$EdgesWalk[E(W),W,G] := (Walk[W,G]) \wedge (E(W) = \{\{w_i, w_{i+1}\} \mid i \in \mathbb{N}_1^{|W|-1}\})$
$Trail[W,G] := (Walk[W,G]) \wedge (\forall_{i,j \in \mathbb{N}_1^{|W|-1}}((i \neq j) \implies (\{w_i, w_{i+1}\} \neq \{w_j, w_{j+1}\})))$
$uvWalk[(u,v),W,G] := (Walk[W,G]) \wedge (W_1 = u) \wedge (W_{|W|} = v)$
$uvTrail[(u,v),W,G] := (Trail[W,G]) \wedge (W_1 = u) \wedge (W_{|W|} = v)$
$uvPath[(u,v),P] := (PathG[P]) \wedge (u,v \in V(P)) \wedge (d(u) = 1 = d(v))$
$LengthWalk[e(W),W,G] := (Walk[W,G]) \wedge (e(W) = |E(W)|)$
$ClosedWalk[W,G] := (Walk[W,G]) \wedge (w_1 = w_{|W|})$
$OddWalk[W,G] := (Walk[W,G]) \wedge (Odd(e(W)))$
$EvenWalk[W,G] := (Walk[W,G]) \wedge (Even(e(W)))$
$WalkContainsPath[P,W,G] := (Path[P]) \wedge (Walk[W,G]) \wedge (OrderedSublist[V(P),W]) \wedge (OrderedSublist[E(P),E(W)])$
$WalkContainsCycle[C,W,G] := (Cycle[C]) \wedge (Walk[W,G]) \wedge (OrderedSublist[V(C),W]) \wedge (OrderedSublist[E(C),E(W)])$

$uvWalkContainsuvPath := (uvWalk[(x,y),W,G]) \implies (\exists_P((uvPath[(x,y),P]) \wedge (WalkContainsPath[P,W,G])))$

(1) $(e(W) = 0) \implies (P = (W,\emptyset))$ ∎ $WalkContainsPath[P,W,G]$

(2) $((e(W) > 0) \land (\forall_{W'}((e(W') < e(W)) \implies ((uvWalk[(x,y),W',G]) \implies (\exists_{P'}((uvPath[(x,y),P']) \land (WalkContainsPath[P',W',G])))))) \implies \ldots$

(2.1) If W has no duplicate vertices, then P = W ▌ $WalkContainsPath[P,W,G]$

(2.2) If W has duplicate vertices, then delete the duplicate vertices and edges between extra copies of unique vertices. This shorter uvWalk W' has a uvPath P' by IH. ▌ $WalkContainsPath[P',W,G]$

(3) $((e(W) > 0) \land (\forall_{W'}((e(W') < e(W)) \implies ((uvWalk[(x,y),W',G]) \implies (\exists_{P'}((uvPath[(x,y),P']) \land (WalkContainsPath[P',W',G])))))) \implies (WalkContainsPath[P,W,G])$

(4) By induction: $(uvWalk[(x,y),W,G]) \implies (\exists_P((uvPath[(x,y),P]) \land (WalkContainsPath[P,W,G])))$

$ConnectedV[(x,y),G] := \exists_P((Subgraph[P,G]) \land (uvPath[(x,y),P]))$
$Connected[G] := \forall_{x,y \in V(G)}(ConnectedV[(x,y),G])$
$Connection[C_G,G] := C_G = \{\langle x,y\rangle \mid ConnectedV[(x,y),G]\}$

$$ConnectionEqRel := \forall_G \forall_{x_1,x_2,x_3 \in G}\left(\begin{array}{l} (x_1 C_G x_1) \qquad\qquad\qquad \land \\ ((x_1 C_G x_2) \implies (x_2 C_G x_1)) \qquad \land \\ (((x_1 C_G x_2) \land (x_2 C_G x_3)) \implies (x_1 \cong x_3)) \end{array}\right)$$

(1) By $(uvWalkContainsuvPath) \land (uvPath[(x,y),W]) \iff (uvPath[(y,x),W])$

$ConnectedSubgraph[H,G] := (Subgraph[H,G]) \land (Connected[H])$
$Component[H,G] := ConnectedSubgraph[H,G] \land (\neg\exists_{K \neq H}((Subgraph[H,K]) \land (ConnectedSubgraph[K,G])))$
$Trivial[G] := E(G) = \emptyset$
$Isolated[v,G] := d(v) = 0$

$Components[\mathcal{H},G] := Partition[\mathcal{H},G,C_G]$
$NumComponents[c,G] := (Components[\mathcal{H},G]) \land (c = |\mathcal{H}|)$

$NumComponentsBound := ((|V(G)| = n) \land (|E(G)| = k)) \implies (n - k \leq |\mathcal{H}|)$

(1) Starting from $E(G) = \emptyset$, $|\mathcal{H}| = n$

(2) Adding an edge would decrease the number of components by 0 or 1, so after adding $k$ edges, $n - k \leq |\mathcal{H}|$

$RemoveV[G-W,W,G] := (V(G-W) = V(G) \setminus W) \land (E(G-W) = \{\{x,y\} \in E(G) \mid x,y \in V(G-W)\})$
$RemoveE[G-E,E,G] := (V(G-E) = V(G)) \land (E(G-E) = E(G) \setminus E)$
$AddE[G+e,e,G] := (e \in V(G)^{\{2\}}) \land (V(G+e) = V(G)) \land (E(G+e) = E(G) \cup \{e\})$
$InducedSubgraph[G[T],T,G] := G[T] = G - \bar{T}$
$IndependentSet[S,G] := E(G[S]) = \emptyset$

$CutVertex[v,G] := (NumComponents[c_1,G]) \land (NumComponents[c_2,G-v]) \land (c_2 > c_1)$
$CutEdge[e,G] := (NumComponents[c_1,G]) \land (NumComponents[c_2,G-e]) \land (c_2 > c_1)$

$CutEdgeEquiv := (CutEdge[e,G]) \iff (\neg\exists_C((Subgraph[C,G]) \land (CycleG[C]) \land (e \in E(C))))$

(1) Let $(Component[H,G]) \land (e = \{x,y\} \in E(H))$

(2) $(CutEdge[e,G])) \iff (CutEdge[e,H])) \iff (\neg Connected[H-e])$

(3) WTS: $(Connected[H-e]) \iff (\exists_C((CycleG[C]) \land (Subgraph[C,G]) \land (e \in E(C))))$

(4) $(Connected[H-e]) \implies \ldots$

(4.1) $\exists_P((PathG[P]) \land (Subgraph[P,H-e]))$ ▌ $CycleG[(V(P),E(P) \cup \{e\})]$ ▌ $\exists_C(((CycleG[C]) \land Subgraph[C,G]) \land (e \in E(C)))$

(5) $(Connected[H-e]) \implies (\exists_C((CycleG[C]) \land (Subgraph[C,G]) \land (e \in E(C))))$

(6) $(\exists_C((CycleG[C]) \land (Subgraph[C,G]) \land (e \in E(C)))) \implies \ldots$

(6.1) $Component[H,G]$ ▌ $Connected[H]$

(6.2) $(u,v \in V(H)) \implies \ldots$

(6.2.1) $\exists_P((Subgraph[P,H]) \land (uvPath[(u,v),P]))$

(6.2.2) $(e \notin E(P)) \implies \ldots$

(6.2.2.1) $(Subgraph[P,H-e])$ ▌ $\exists_P((Subgraph[P,H-e]) \land (uvPath[(u,v),P]))$

(6.2.3) $(e \notin E(P)) \implies (\exists_P((Subgraph[P,H-e]) \land (uvPath[(u,v),P])))$

(6.2.4) $(e \in E(P)) \implies \ldots$

(6.2.4.1) $P' = u - xPath + x - yCycleG + y - vPath$

(6.2.4.2) $(Subgraph[P', H - e]) \land (uvPath[(u, v), P'])$ ▮ $\exists_P((Subgraph[P, H - e]) \land (uvPath[(u, v), P]))$

(6.2.5) $(e \in E(P)) \implies (\exists_P((Subgraph[P, H - e]) \land (uvPath[(u, v), P])))$

(6.2.6) $\exists_P((Subgraph[P, H - e]) \land (uvPath[(u, v), P]))$

(6.3) $(u, v \in V(H)) \implies (\exists_P((Subgraph[P, H - e]) \land (uvPath[(u, v), P])))$ ▮ $Connected[H - e]$

(7) $(\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (e \in E(C)))) \implies (Connected[H - e])$

(8) $(Connected[H - e]) \iff (\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (e \in E(C))))$

$COWalkContainsOCycle := ((ClosedWalk[W, G]) \land (OddWalk[W, G])) \implies (\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C)))))$

(1) $(e(W) = 1) \implies (C = (\{w_1\}, \emptyset))$ ▮ $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

(2) $((e(W) > 1) \land (\forall_{W'}((e(W') < e(W)) \implies$
$(((ClosedWalk[W', G]) \land (OddWalk[W', G])) \implies (\exists_{C'}((WalkContainsCycle[C', W', G]) \land (Odd(e(C'))))))))) \implies \ldots$

(2.1) If $W$ has no repeated vertex other than the first and last, then $C = (W, E(W))$ ▮ $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

(2.2) If $W$ has a repeated vertex $v$, then ...

(2.2.1) Break $W$ into two $v$ Walks $W_1, W_2$. Since $W$ is odd, $W_1, W_2$ are odd and even walks (not in order).

(2.2.2) WLOG let $W_1$ be the odd subwalk, then by IH $\exists_{C'}((WalkContainsCycle[C', W_1, G]) \land (Odd(e(C'))))$

(2.2.3) $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

(2.3) If $W$ has a repeated vertex $v$, then $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

(2.4) $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

(3) $((e(W) > 1) \land (\forall_{W'}((e(W') < e(W)) \implies$
$(((ClosedWalk[W', G]) \land (OddWalk[W', G])) \implies (\exists_{C'}((WalkContainsCycle[C', W', G]) \land (Odd(e(C'))))))))) \implies$
$(\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C)))))$

(4) By induction: $\exists_C((WalkContainsCycle[C, W, G]) \land (Odd(e(C))))$

$Bipartiton[\{X, Y\}, G] := PartiteSets[\{X, Y\}, G]$
$ConnectedBipartite[G] := \exists!_{\{X,Y\}}(Bipartiton[\{X, Y\}, G])$

$BipartiteEquiv := (Bipartite[G]) \iff (\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C)))))$

(1) $(Bipartite[G]) \implies \ldots$

(1.1) Every step alternates between each bipartition. Thus the end vertex of the odd walk cannot be the start vertex, and it is not a cycle.

(1.2) $\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C))))$

(2) $(Bipartite[G]) \implies (\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C)))))$

(3) $(\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C))))) \implies \ldots$

(3.1) Consider each nontrivial component $H$, and pick a $u \in V(H)$.

(3.2) Let $X = \{v \in H \mid Even(d(v, u))\}$ and let $Y = \{v \in H \mid Odd(d(v, u))\}$.

(3.3) Suppose $X$ or $Y$ are not independent sets. WLOG choose $X$.

(3.3.1) $X$ must contain an edge - call it $\{v, v'\}$

(3.3.2) A closed odd walk could be: min u-v path (+ even) and v-v' (+ 1) and min v'-u path (+ even)

(3.3.3) By $COWalkContainsOCycle$, there exists an odd cycle in $G$. ▮ $\bot$

(3.4) $X$ and $Y$ are independent sets; futhermore X, Y are bipartitions of G. ▮ $Bipartite[G]$

(4) $(\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C))))) \implies (Bipartite[G])$

(5) $(Bipartite[G]) \iff (\neg\exists_C((CycleG[C]) \land (Subgraph[C, G]) \land (Odd(e(C)))))$

$UnionG[\cup(\mathcal{G}), \mathcal{G}] := (V(\cup(\mathcal{G})) = \bigcup_{G \in \mathcal{G}} (V(G))) \land (E(\cup(\mathcal{G})) = \bigcup_{G \in \mathcal{G}} (E(G)))$

$CompleteAsBipartiteUnion := (\exists_{\langle B \rangle_1^k}((\forall_{B \in \langle B \rangle_1^k}(BipartiteG[B])) \land (UnionG[K_n, \langle B \rangle_1^k]))) \iff (n \leq 2^k)$

(1) $(k = 1) \implies \ldots$

(1.1) $(\exists_{\langle B \rangle_1^k}((\forall_{B \in \langle B \rangle_1^k}(BipartiteG[B])) \land (UnionG[K_n, \langle B \rangle_1^k]))) \iff (Bipartite[K_n])$

(1.2) $(n \leq 2^k) \implies \ldots$

(1.2.1) $n \leq 2^1 = 2$ ▮ $((n = 1) \lor (n = 2))$

(1.2.2) $(BipartiteG[K_1]) \land (BipartiteG[K_2])$ ▮ $Bipartite[K_n]$

(1.3) $(n \leq 2^k) \implies (Bipartite[K_n])$

(1.4) $(Bipartite[K_n]) \implies \ldots$

(1.4.1) $(n > 2) \implies \ldots$

(1.4.1.1) $K_n$ has an odd cycle

(1.4.1.2) $BipartiteEquiv$ and $K_n$ has an odd cycle $\blacksquare$ $\neg Bipartite[K_n]$ $\blacksquare$ $\bot$

(1.4.2) $(n > 2) \implies (\bot)$ $\blacksquare$ $n \leq 2$

(1.5) $(Bipartite[K_n]) \implies (n \leq 2)$

(1.6) $(Bipartite[K_n]) \iff (n \leq 2)$ $\blacksquare$ $(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2)$

(2) $(k = 1) \implies ((\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2))$

(3) $((k > 1) \wedge (\forall_{k'}((k' < k) \implies ((\exists_{\langle B\rangle_1^{k'}}((\forall_{B\in\langle B\rangle_1^{k'}}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^{k'}]))) \iff (n \leq 2^{k'}))))) \implies \ldots$

(3.1) $(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \implies \ldots$

(3.1.1) $K_n = \cup(\langle B\rangle_1^k) = \bigcup_{i=1}^{k}(B_i) = \bigcup_{i=1}^{k-1}(B_i) \cup B_k$ $\blacksquare$ $K_n = \bigcup_{i=1}^{k-1}(B_i) \cup B_k$

(3.1.2) $Bipartite[B_k]$ $\blacksquare$ $\exists_{X_0,Y_0}(PartiteSets[\{X_0, Y_0\}, B_k])$ $\blacksquare$ $\exists_{X,Y}(PartiteSets[\{X, Y\}, (V(G), E(B_k))])$

(3.1.3) $K_n = (\bigcup_{i=1}^{k-1}(B_i) \cup B_k) \wedge (PartiteSets[\{X, Y\}, B_k])$ $\blacksquare$ $\bigcup_{i=1}^{k-1}(B_i) = K_n[X] \cup K_n[Y]$

(3.1.4) $\bigcup_{i=1}^{k-1}(B_i) = K_n[X] \cup K_n[Y]$ and IH $\blacksquare$ $(|X| = n(K_n[X]) \leq 2^{k-1}) \wedge (|Y| = n(K_n[Y]) \leq 2^{k-1})$

(3.1.5) $n = |G| = |X| + |Y| \leq 2^{k-1} + 2^{k-1} = 2^k$ $\blacksquare$ $n \leq 2^k$

(3.2) $(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \implies (n \leq 2^k)$

(3.3) $(n \leq 2^k) \implies \ldots$

(3.3.1) $\exists_{X,Y}((X \dot\cup Y = V(K_n)) \wedge (|X| \leq 2^{k-1}) \wedge (|Y| \leq 2^{k-1}))$

(3.3.2) IH $\blacksquare$ $(\exists_{\langle X\rangle_1^{k-1}}((\forall_{X\in\langle X\rangle_1^{k-1}}(BipartiteG[X])) \wedge (UnionG[K_n[X], \langle X\rangle_1^{k-1}]))) \wedge$
$(\exists_{\langle Y\rangle_1^{k-1}}((\forall_{Y\in\langle Y\rangle_1^{k-1}}(BipartiteG[Y])) \wedge (UnionG[K_n[Y], \langle Y\rangle_1^{k-1}])))$

(3.3.3) $(\langle Z\rangle_1^{k-1} = \langle X_i \cup Y_i\rangle_{i=1}^{k-1}) \wedge (CompleteBipartiteG[Z_k, X, Y])$ $\blacksquare$ $(\forall_{Z\in\langle Z\rangle_1^k}(BipartiteG[Z])) \wedge (UnionG[K_n, \langle Z\rangle_1^k])$

(3.4) $(n \leq 2^k) \implies (\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k])))$

(3.5) $(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2^k)$

(4) $((k > 1) \wedge (\forall_{k'}((k' < k) \implies ((\exists_{\langle B\rangle_1^k}((|\langle B\rangle_1^k| = k') \wedge (\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2^{k'}))))) \implies$
$(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2)$

(5) By induction: $(\exists_{\langle B\rangle_1^k}((\forall_{B\in\langle B\rangle_1^k}(BipartiteG[B])) \wedge (UnionG[K_n, \langle B\rangle_1^k]))) \iff (n \leq 2)$

$Circuit[W, G] := (Trail[W, G]) \wedge (ClosedWalk[W, G])$
$EulerianTrail[W, G] := ((Trail[W, G])) \wedge (E(W) = E(G))$
$EulerianCircuit[W, G] := ((Circuit[W, G])) \wedge (E(W) = E(G))$
$Eulerian[G] := \exists_W(EulerianCircuit[W, G])$

$OddVertex[v, G] := Odd(d(v))$
$EvenVertex[v, G] := Even(d(v))$
$EvenGraph[G] := \forall_{v\in V(G)}(EvenVertex[v, G])$

$MaximalPath[P, G] := (Subgraph[P, G]) \wedge (PathG[P]) \wedge (\neg\exists_{P'\neq P}((Subgraph[P, P']) \wedge (Subgraph[P', G]) \wedge (PathG[P'])))$
$MaximalTrail[W, G] := (Trail[W, G]) \wedge (\neg\exists_{W'\neq W}((W \subseteq W') \wedge (Trail[W', G])))$

$VertexDegreeCycle := (\forall_{v\in V(G)}(2 \leq d(v))) \implies (\exists_C((Subgraph[C, G]) \wedge (CycleG[C])))$

(1) $\exists_P(MaximalPath[P, G])$ $\blacksquare$ $\exists_{u,v}(uvPath[(u, v), P])$

(2) Since $P$ is maximal, adjacent vertices of $u$ must be contained in $P$.

(3) Since $2 \leq d(u)$, then $u$ has at least 2 edges that are incident among the vertices in $P$.

(4) These edges form a cycle from $u$. $\exists_C((Subgraph[C, G]) \wedge (CycleG[C]))$.

$EulerianEquiv := (Components[\mathcal{H}, G]) \implies ((Eulerian[G]) \iff (((\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])) \wedge (EvenGraph[G])))$

(1) $(Eulerian[G]) \implies \ldots$

(1.1) $Eulerian[G]$ ▌ $\exists_W(EulerianCircuit[W, G])$

(1.2) The first and last vertices have even degree, and the intermediate vertices have even degree. ▌ $EvenGraph[G]$

(1.3) $E(G)$ must be covered by the $W$, thus they must lie on the same non-trivial component. ▌ $(\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])$

(1.4) $((\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])) \wedge (EvenGraph[G])$

(2) $(Eulerian[G]) \implies (((\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])) \wedge (EvenGraph[G]))$

(3) $(((\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])) \wedge (EvenGraph[G])) \implies \ldots$

(3.1) $(E(G) = 0) \implies \ldots$

(3.1.1) Let the Eulerian circuit be consist of just one vertex. ▌ $Eulerian[G]$

(3.2) $(E(G) = 0) \implies (Eulerian[G])$

(3.3) $((E(G) > 0) \wedge (\forall_{G'}((E(G') < E(G)) \implies (Eulerian[G'])))) \implies \ldots$

(3.3.1) $\exists!_H(H \in \mathcal{H} \mid \neg Trivial[H])$

(3.3.2) $EvenGraph[G]$ ▌ $EvenGraph[H]$ ▌ $\forall_{v \in V(H)}(2 \leq d(v))$

(3.3.3) $VertexDegreeCycle$ ▌ $\exists_C((Subgraph[C, H]) \wedge (CycleG[C]))$

(3.3.4) $G' := G - E(C)$

(3.3.5) Since the vertices in a cycle have degree 2, $EvenGraph[G']$. Each $H'$ component of $G'$ is also an $EvenGraph[H']$.

(3.3.6) By IH and $\forall_{H' \in \mathcal{H}'}(E(H') < E(G))$ ▌ $\forall_{H' \in \mathcal{H}'}(Eulerian[H'])$

(3.3.7) The Eulerian circuit of $G$ can be constructed by:

(3.3.7.1) Start at some vertex in $C$

(3.3.7.2) Go around $C$, until the trail reaches a vertex of some $H' \in \mathcal{H}'$

(3.3.7.3) Trail around $H'$ using it's own Eulerian trail, and return to the vertex in $C'$.

(3.3.7.4) Continue the last two steps until the trail of $C$ is complete.

(3.3.8) $Eulerian[G]$

(3.4) $((E(G) > 0) \wedge (\forall_{G'}((E(G') < E(G)) \implies (Eulerian[G'])))) \implies ((Eulerian[G]))$

(4) $(((\nexists \vee \exists!)_{H \in \mathcal{H}}(\neg Trivial[H])) \wedge (EvenGraph[G])) \implies (Eulerian[G])$

$EvenGraphCycles := (EvenGraph[G]) \implies (\exists_{\mathcal{D}}((Decomposition[\mathcal{D}, G]) \wedge (\forall_{D \in \mathcal{D}}(Cycle[D]))))$

(1) $(E(G) = 0) \implies \ldots$

(1.1) $\mathcal{D} = \{G\}$ ▌ $\exists_{\mathcal{D}}((Decomposition[\mathcal{D}, G]) \wedge (\forall_{D \in \mathcal{D}}(Cycle[D])))$

(2) $((E(G) > 0) \wedge (\forall_{G'}((E(G') < E(G)) \implies ((EvenGraph[G']) \implies (\exists_{\mathcal{D}'}((Decomposition[\mathcal{D}', G']) \wedge (\forall_{D' \in \mathcal{D}'}(Cycle[D']))))))))$
$\implies \ldots$

(2.1) $(E(G) > 0) \wedge (EvenGraph[G])$ ▌ $\forall_{v \in V(G)}(2 \leq d(v))$

(2.2) $VertexDegreeCycle$ ▌ $\exists_C((Subgraph[C, G]) \wedge (CycleG[C]))$

(2.3) $G' := G - E(C)$

(2.4) Since the vertices in a cycle have degree 2, $EvenGraph[G']$. Each $D'$ component of $G'$ is also an $EvenGraph[D']$.

(2.5) $E(D') < E(G)$ and IH, there exists a cycle decomposition of $D'$.

(2.6) The cycle decomposition of G can be constructed by collecting the cycle decompositions of all $D' \in \mathcal{D}'$ and including $C$.

(2.7) $\exists_{\mathcal{D}}((Decomposition[\mathcal{D}, G]) \wedge (\forall_{D \in \mathcal{D}}(Cycle[D])))$

(3) $((E(G) > 0) \wedge (\forall_{G'}((E(G') < E(G)) \implies ((EvenGraph[G']) \implies (\exists_{\mathcal{D}'}((Decomposition[\mathcal{D}', G']) \wedge (\forall_{D' \in \mathcal{D}'}(Cycle[D']))))))))$
$\implies (\exists_{\mathcal{D}}((Decomposition[\mathcal{D}, G]) \wedge (\forall_{D \in \mathcal{D}}(Cycle[D]))))$

(4) By induction, $\exists_{\mathcal{D}}((Decomposition[\mathcal{D}, G]) \wedge (\forall_{D \in \mathcal{D}}(Cycle[D])))$

$VertexDegreePathk := (\forall_{v \in V(G)}(k \leq d(v))) \implies (\exists_P((Subgraph[P, G]) \wedge (PathG[P]) \wedge (k \leq e(P))))$

(1) $\exists_P(MaximalPath[P, G])$ ▌ $\exists_{u,v}(uvPath[(u, v), P])$

(2) Since $P$ is maximal, adjacent vertices of $u$ must be contained in $P$.

(3) Since $k \leq d(u)$, then $u$ has at least $k$ edges that are incident among the vertices in $P$.

(4) Thus $P$ has at least $k$ vertices. ▌ $k \leq E(P)$.

(5) $\exists_P((Subgraph[P, G]) \wedge (PathG[P]) \wedge (k \leq e(P)))$

$VertexDegreeCyclek := ((k \geq 2) \wedge (\forall_{v \in V(G)}(k \leq d(v)))) \implies (\exists_C((Subgraph[C, G]) \wedge (CycleG[C]) \wedge (k + 1 \leq e(C))))$

(1) $VertexDegreePathk$ ▌ $\exists_P((Subgraph[P, G]) \wedge (PathG[P]) \wedge (k \leq e(P)))$

(2) The edge formed by $u$ and it's farthest neighbor along $P$ will form a cycle $C$ with $k+1 \leq e(C)$

(3) $((k \geq 2) \wedge (\forall_{v \in V(G)}(k \leq d(v)))) \implies (\exists_C((Subgraph[C,G]) \wedge (CycleG[C]) \wedge (k+1 \leq e(C))))$

$NonCutVertices := (n(G) \geq 2) \implies (\exists_{x,y \in V(G)}((x \neq y) \wedge (\neg CutVertex[x,G]) \wedge ((\neg CutVertex[y,G]))))$

(1) $\exists_P(MaximalPath[P,G])$ ▌ $\exists_{u,v}(uvPath[(u,v),P])$

(2) $Connected[P-u]$ ▌ $\neg CutVertex[u,G]$

(3) $(v \neq u) \implies (\neg CutVertex[v,G])$

(4) $(v = u) \implies \ldots$ ▌ Take another maximal path within $P-u$. ▌ Take another endpoint $u'$. ▌ $\neg CutVertex[u',G]$

$EvenGraphMaximalTrailClosed := ((EvenGraph[G]) \wedge (MaximumTrail[W,G])) \implies (ClosedWalk[W,G])$

(1) Every step in $W$ adds 1 degree to each endpoint.

(2) Thus when arriving at a vertex $u$ that is not the initial vertex, $u$ will have an odd count of edges incident to it.

(3) Since $u$ has an even degree, then there remains an edge where $W$ can continue.

(4) Therefore, the $W$ can only end (become maximal) when it reaches it's initial vertex. ▌ $ClosedWalk[W,G]$

$OddVertexTrailDecomposition := ((Connected[G]) \wedge (|\{v \in V(G) \mid Odd(d(v))\}| = 2k))$
$\implies (\exists_{\mathcal{D}}((\forall_{D \in \mathcal{D}}(Trail[D,G])) \wedge (Decomposition[\mathcal{D},G]) \wedge (|\mathcal{D}| = max(\{k,1\}))))$

(1) $(k = 0) \implies \ldots$

(1.1) $k = 0$ ▌ $EvenGraph[G]$

(1.2) $Connected[G]$ ▌ $\exists!_{H \in \mathcal{H}}(\neg Trivial[H])$

(1.3) $EulerianEquiv$ ▌ $Eulerian[G]$ ▌ $\exists_W(EulerianCircuit[W,G])$

(1.4) $D := (V(G), E(W))$ ▌ $(Trail[D,G]) \wedge (Decomposition[\{D\},G]) \wedge (\{D\} = 1 = max(\{k,1\}))$

(2) $(k = 0) \implies (\exists_{\mathcal{D}}((\forall_{D \in \mathcal{D}}(Trail[D,G])) \wedge (Decomposition[\mathcal{D},G]) \wedge (|\mathcal{D}| = max(\{k,1\}))))$

(3) $(k > 0) \implies \ldots$

(3.1) Since each trail adds an even degree to each non-endpoint vertex, we need at least $k$ trails to partition the $2k$ odd vertices.

(3.2) Partition the edges into $k$ trails such that the ends of each trail will land on an odd vertex.

(3.3) Construct a new graph $G'$ where the $k$ trails are connected by an edge. ▌ $(\exists!_{H' \in \mathcal{H}'}(\neg Trivial[H'])) \wedge (EvenGraph[G'])$

(3.4) $EulerianEquiv$ ▌ $Eulerian[G']$ ▌ $\exists_{W'}(EulerianCircuit[W',G'])$

(3.5) Construct $\mathcal{D}$ to be the trails in $W'$ separated by $E(G) \setminus E(G')$. ▌ $(Decomposition[\mathcal{D},G]) \wedge (\mathcal{D} = k))$

(4) $(k > 0) \implies (\exists_{\mathcal{D}}((\forall_{D \in \mathcal{D}}(Trail[D,G])) \wedge (Decomposition[\mathcal{D},G]) \wedge (|\mathcal{D}| = max(\{k,1\}))))$

(5) $\exists_{\mathcal{D}}((\forall_{D \in \mathcal{D}}(Trail[D,G])) \wedge (Decomposition[\mathcal{D},G]) \wedge (|\mathcal{D}| = max(\{k,1\})))$

### 2.1.4 Vertex Degrees and Counting

$MinDegree[\delta(G),G] := \delta(G) = min(\{d(v) \mid v \in V(G)\})$
$MinDegree[\Delta(G),G] := \Delta(G) = max(\{d(v) \mid v \in V(G)\})$
$RegularG[G] := \delta(G) = \Delta(G)$
$kRegularG[G,k] := k = \delta(G) = \Delta(G)$
$Neighborhood[N(v),v,G] := N(v) = \{u \in V(G) \mid AdjacentV[\{u,v\},G]\}$

$DegreeSumFormula := \sum_{v \in V(G)}(d(v)) = 2e(G)$

(1) $\sum_{v \in V(G)}(d(v)) = \sum_{v \in V(G)}(|\{e \in E(G) | v \in e\}|) = 2|E(G)| = 2e(G)$

$AverageDegree := \delta(G) \leq \frac{2e(G)}{n(G)} \leq \Delta(G)$

(1) $\delta(G) \leq \frac{2e(G)}{n(G)} \leq \Delta(G)$

$EvenNumberOfOddVertices := Even(|\{v \in V(G) \mid Odd(d(v))\}|)$

(1) $DegreeSumFormula$ ▌ $Even(\sum_{v \in V(G)}(d(v)))$

(2) $(Odd(|\{v \in V(G) \mid Odd(d(v))\}|)) \implies (Odd(\sum_{v \in V(G)}(d(v)))) \implies (\perp)$ ▮ $Even(|\{v \in V(G) \mid Odd(d(v))\}|)$

$kRegularGraphSize := ((kRegularG[G,k]) \wedge (n(G) = n)) \implies (e(G) = nk/2)$

(1) $DegreeSumFormula$ ▮ $2e(G) = \sum_{i=1}^{n}(d(v_i)) = \sum_{i=1}^{n}(k) = nk$ ▮ $e(G) = nk/2$

$kCube[Q_k, k] := (V(Q_k) = \{0,1\}^k) \wedge (E(Q_k) = \{\{x,y\} \mid diff(x,y) = 1\})$

$RegularPartiteSetSize := ((k > 0) \wedge (kRegularG[G,k]) \wedge (Bipartiton[\{X,Y\},G])) \implies (|X| = |Y|)$

(1) $kRegularG[G,k]$ ▮ $(e(G) = 2|X|) \wedge (e(G) = 2|Y|)$ ▮ $|X| = |Y|$

### 2.1.5 Trees

$Acyclic[G] := \neg\exists_C((Subgraph[C,G]) \wedge (CycleG[C]))$
$Forest[G] := Acyclic[G]$
$Tree[G] := (Connected[G]) \wedge (Acyclic[G])$
$Leaf[v,G] := d(v) = 1$
$SpanningSubgraph[H,G] := (Subgraph[H,G]) \wedge (V(H) = V(G))$
$SpanningTree[H,G] := (SpanningSubgraph[H,G]) \wedge (Tree[G])$

$LeafExistence := ((Tree[G]) \wedge (2 \leq n(G))) \implies (2 \leq |\{v \in V(G) \mid Leaf[v,G]\}|)$

(1) $Tree[G]$ ▮ $(Connected[G]) \wedge (Acyclic[G])$

(2) $(2 \leq n(G)) \wedge (Connected[G])$ ▮ $\exists_e(e \in E(G))$ ▮ Let $P$ be the maximal path of $e$.

(3) A maximal non-trivial path with no cycles has two endpoints. ▮ $2 \leq |\{v \in V(G) \mid Leaf[v,G]\}|$

$LeafDeletion := ((Tree[G]) \wedge (n(G) = n) \wedge (Leaf[v,G])) \implies ((Tree[G-v]) \wedge (n(G-v) = n-1))$

(1) $Tree[G]$ ▮ $(Connected[G]) \wedge (Acyclic[G])$

(2) Since $d(v) = 1$, $v$ does not belong to any path connecting any other two $u_1, u_2 \in V(G)$. ▮ $Connected[G-v]$

(3) Since deleting a vertex cannot create a cycle. ▮ $Acyclic[G-v]$

(4) $Tree[G-v]$

$$TreeEquiv := (n = n(G) \geq 1) \implies \begin{pmatrix} (A) & (Tree[G]) & \iff \\ (B) & ((Connected[G]) \wedge (e(G) = n-1)) & \iff \\ (C) & ((Acyclic[G]) \wedge (e(G) = n-1)) & \iff \\ (D) & (\forall_{u,v \in V(G)}\exists!_P(uvPath[(u,v),P])) & \end{pmatrix}$$

(1) $(Tree[G]) \implies \ldots [A \implies B]$

(1.1) $Tree[G]$ ▮ $Connected[G]$

(1.2) $(n = 1) \implies (e(G) = 0 = n-1)$

(1.3) $((n > 1) \wedge (\forall_{G'}(((n(G') < n) \wedge (Tree[G'])) \implies (e(G') = n(G') - 1)))) \implies \ldots$

(1.3.1) $LeafExistence$ ▮ $\exists_{v \in V(G)}(Leaf[v,G])$

(1.3.2) $LeafDeletion$ ▮ $Tree[G-v]$

(1.3.3) By IH, $e(G-v) = (n-1) - 1 = n-2$

(1.3.4) $Leaf[v,G]$ ▮ $e(G) = e(G-v) + 1 = n-1$

(1.4) $((n > 1) \wedge (\forall_{G'}(((n(G') < n) \wedge (Tree[G'])) \implies (e(G') = n(G') - 1)))) \implies (e(G) = n-1)$

(1.5) By induction, $e(G) = n-1$ ▮ $(Connected[G]) \wedge (e(G) = n-1)$

(2) $(Tree[G]) \implies ((Connected[G]) \wedge (e(G) = n-1))$

(3) $((Connected[G]) \wedge (e(G) = n-1)) \implies \ldots [B \implies C]$

(3.1) Delete all edges that form a cycle in $G$ to form $G'$. ▮ $Acyclic[G']$

(3.2) $(Connected[G]) \wedge (CutEdgeEquiv)$ ▮ $Connected[G']$

(3.3) $(Connected[G']) \wedge (Acyclic[G']) \wedge ([A \implies B])$ ▮ $e(G') = n-1$

(3.4) By construction of $G'$ and $e(G) = n-1 = e(G')$, $G = G'$. ▮ $Acyclic[G]$

(3.5) Equivalently, $G' = G - E = G - \emptyset = G \;\blacksquare\; G = G'$

(3.6) $(Acyclic[G]) \wedge (e(G) = n-1)$

(4) $((Connected[G]) \wedge (e(G) = n-1)) \implies ((Acyclic[G]) \wedge (e(G) = n-1))$

(5) $((Acyclic[G]) \wedge (e(G) = n-1)) \implies \ldots [C \implies A]$

(5.1) $Acyclic[G]$

(5.2) $Components[\langle G_i \rangle_{i=1}^{k}, G] \;\blacksquare\; \sum_{i=1}^{k}(n(G_i)) = n(G) = n$

(5.3) $\forall_{i \in \mathbb{N}_1^k}(Component[G_i, G]) \;\blacksquare\; \forall_{i \in \mathbb{N}_1^k}(Connected[G_i])$

(5.4) $\forall_{i \in \mathbb{N}_1^k}((Connected[G_i]) \wedge (Acyclic[G_i]))$

(5.5) $([A \implies B]) \wedge (\forall_{i \in \mathbb{N}_1^k}((Connected[G_i]) \wedge (Acyclic[G_i]))) \;\blacksquare\; \forall_{i \in \mathbb{N}_1^k}(e(G_i) = n(G_i) - 1)$

(5.6) $e(G) = \sum_{i=1}^{k}(e(G_i)) = \sum_{i=1}^{k}(n(G_i) - 1) = n - k$

(5.7) $(e(G) = n-k) \wedge (e(G) = n-1) \;\blacksquare\; k = 1 \;\blacksquare\; Connected[G]$

(5.8) $(Connected[G]) \wedge (Acyclic[G]) \;\blacksquare\; Tree[G]$

(6) $((Acyclic[G]) \wedge (e(G) = n-1)) \implies (Tree[G])$

(7) $(Tree[G]) \implies \ldots [A \implies D]$

(7.1) $Tree[G] \;\blacksquare\; (Connected[G]) \wedge (Acyclic[G])$

(7.2) $Connected[G] \;\blacksquare\; \forall_{u,v \in V(G)} \exists_P (uvPath[(u,v), P])$

(7.3) $((u, v \in V(G)) \wedge (uvPath[(u,v), P_1]) \wedge (uvPath[(u,v), P_2])) \implies \ldots$

(7.3.1) $(P_1 \neq P_2) \implies \ldots$

(7.3.1.1) Take the shortest subpaths $P_1', P_2'$ of $P_1, P_2$ that ends on the same endpoints $u', v'$.

(7.3.1.2) By the extremal choice, $P_1', P_2'$ share the same endpoints, but no internal vertices. $\blacksquare\; Cycle[P_1' \cup P_2']$

(7.3.1.3) $(Acyclic[G]) \wedge (Cycle[P_1' \cup P_2']) \;\blacksquare\; \bot$

(7.3.2) $(P_1 \neq P_2) \implies (\bot) \;\blacksquare\; P_1 = P_2$

(7.4) $((u, v \in V(G)) \wedge (uvPath[(u,v), P_1]) \wedge (uvPath[(u,v), P_2])) \implies (P_1 = P_2)$

(8) $(Tree[G]) \implies (\forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P]))$

(9) $(\forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P])) \implies \ldots [D \implies A]$

(9.1) $\forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P]) \;\blacksquare\; \forall_{u,v \in V(G)} \exists_P (uvPath[(u,v), P]) \;\blacksquare\; Connected[G]$

(9.2) $(\neg Acyclic[G]) \implies \ldots$

(9.2.1) $\exists_C (Cycle[C] \wedge (Subgraph[C, G]))$

(9.2.2) $\forall_{c_1, c_2 \in C} \exists_{P,P'} ((P \neq P') \wedge (uvPath[(c_1, c_2), P]) \wedge (uvPath[(c_1, c_2), P']))$

(9.2.3) $(\forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P])) \wedge (\forall_{c_1, c_2 \in C} \exists_{P,P'} ((P \neq P') \wedge (uvPath[(c_1, c_2), P]) \wedge (uvPath[(c_1, c_2), P']))) \;\blacksquare\; \bot$

(9.3) $(\neg Acyclic[G]) \implies (\bot) \;\blacksquare\; Acyclic[G]$

(9.4) $(Connected[G]) \wedge (Acyclic[G])$

(10) $(\forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P])) \implies (Tree[G])$

$$TreeEquivCorollaries := \begin{pmatrix} (A) & ((Tree[G]) \implies (\forall_{e \in E(G)}(CutEdge[e, G]))) & \wedge \\ (B) & ((Tree[G]) \implies (\exists!_C((Cycle[C]) \wedge (Subgraph[C, G+e]))))\wedge & \\ (C) & ((Connected[G]) \implies (\exists_T(SpanningTree[T, G]))) & \end{pmatrix}$$

(1) $(Tree[G]) \implies \ldots [A]$

(1.1) $Tree[G] \;\blacksquare\; Connected[G]$

(1.2) $TreeEquiv \;\blacksquare\; \forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P]) \;\blacksquare\; \forall_{\{u,v\} \in E(G)}(CutEdge[\{u,v\}, G])$

(2) $(Tree[G]) \implies (\forall_{e \in E(G)}(CutEdge[e, G]))$

(3) $(Tree[G]) \implies \ldots [B]$

(3.1) $Tree[G] \;\blacksquare\; Connected[G]$

(3.2) $TreeEquiv \;\blacksquare\; \forall_{u,v \in V(G)} \exists!_P (uvPath[(u,v), P]) \;\blacksquare\; \exists!_C((Cycle[C]) \wedge (Subgraph[C, G+e]))$

(4) $(Tree[G]) \implies (\exists!_C((Cycle[C]) \wedge (Subgraph[C, G+e])))$

(5) $(Connected[G]) \implies \ldots [C]$

(5.1) Delete all edges that form a cycle in $G$ to form $G'$. $\blacksquare\; (Acyclic[G']) \wedge (V(G') = V(G))$

(5.2) $V(G') = V(G) \;\blacksquare\; SpanningSubgraph[G', G]$

| | |
|---|---|
| (5.3) | $(Connected[G]) \land (CutEdgeEquiv) \;\blacksquare\; Connected[G']$ |
| (5.4) | $(Connected[G']) \land (Acyclic[G']) \;\blacksquare\; Tree[G']$ |
| (5.5) | $(SpanningSubgraph[G', G]) \land (Tree[G']) \;\blacksquare\; SpanningTree[G', G] \;\blacksquare\; \exists_T (SpanningTree[T, G])$ |
| (6) | $(Connected[G]) \implies (\exists_T (SpanningTree[T, G]))$ |

CONTHERE p. 69

### 2.1.6 Coloring

$Coloring[\phi, G] := (Function[\phi, V(G), \mathbb{N}])$
$ProperColoring[\phi, G] := (Coloring[\phi, G]) \land (\forall_{\{x,y\} \in E(G)} (\phi(x) \neq \phi(y)))$
$kColoring[\phi, k, G] := (Coloring[\phi, G]) \land (|\phi(V(G))| = k)$
$kAcceptableColoring[\phi, k, G] := (ProperColoring[\phi, G]) \land (kColoring[\phi, k, G])$
$kColorable[G, k] := \exists_\phi (kAcceptableColoring[\phi, k, G])$
$ChromaticNumber[\chi(G), G] := min(\{k \in \mathbb{N} \mid kColorable[G, k]\})$

$ListAssignment[L, G] := Function[L, V(G), \mathcal{P}(\mathbb{N})]$
$TotalColors[C, L, G] := (ListAssignment[L, G]) \land (C = \bigcup_{v \in V(G)} (L(v)))$
$LColoring[\phi, L, G] := (Coloring[\phi, G]) \land (ListAssignment[L, G]) \land (\forall_{v \in V(G)} (\phi(v) \in L(v))$
$LAcceptableColoring[\phi, L, G] := (ProperColoring[\phi, G]) \land (LColoring[\phi, L, G])$
$LColorable[G, L] := \exists_\phi (LAcceptableColoring[\phi, L, G])$
$kChoosable[G, k] := \forall_L (((ListAssignment[L, G]) \land (\forall_{v \in V(G)} (|L(v)| = k))) \implies (LColorable[G, L]))$
$ListChromaticNumber[\chi_l(G), G] := min(\{k \in \mathbb{N} \mid kChoosable[G, k]\})$

$ChromaticChoosable[G] := \chi_l(G) = \chi(G)$
$PartialLAcceptableColoring[\phi, L, G, A] := (\emptyset \neq A \subseteq V(G)) \land (LAcceptableColoring[\phi, L, G[A]])$

### 2.1.7 Scratch

$Ohba := (n(G) \leq 2\chi(G) + 1) \implies (ChromaticChoosable[G])$
$OhbaEquiv := ((n(G) \leq 2k + 1) \land (CompletekPartite[G, k])) \implies (\chi_l(G) = k = \chi(G))$

$G := min_{|V(G)|} \mid ((n(G) \leq 2k + 1) \land (CompletekPartite[G, k])) \land (\chi_l(G) > k = \chi(G))$
$L := ((ListAssignment[L, G]) \land (\forall_{v \in V(G)} (|L(v)| = k))) \land (\neg LColorable[G, L])$
$C := TotalColors[C, L, G]$