International Journal of Bifurcation and Chaos, Vol. 18, No. 2 (2008) 407–427
© World Scientific Publishing Company

A POLYNOMIAL TIME ALGORITHM TO DETERMINE MAXIMAL BALANCED EQUIVALENCE RELATIONS

JOHN W. ALDIS
Mathematics Institute, University of Warwick,
Coventry CV4 7AL, UK

Received January 24, 2007; Revised August 20, 2007

Following Golubitsky, Stewart, and others, we give definitions of networks and input trees. In order to make our work as general as possible, we work with a somewhat extended notion of multiplicity, and introduce the concept of "bunching" of trees. We then define balanced equivalence relations on networks, and a partial ordering on these relations. Previous work has shown that there is a maximal balanced equivalence relation on networks of certain classes: we provide a different style of proof which gives this result for any network. We define two algorithms to determine this relation in practice on a given finite network — one for use with networks with all multiplicities equal, and a second for the more general case. We then provide illustrative examples of each algorithm in use. We show both of these algorithms to be quartic in the size of the given network.

Keywords: Network; coupled cell; balance; lattice; polynomial time algorithm.

1. Introduction

As described in [Stewart *et al.*, 2003; Golubitsky *et al.*, 2005], a coupled cell network is, informally, a network of nodes or *cells*, each of which has a value associated with it (typically a finite-dimensional real vector). These cells are connected or "coupled" together by edges. One application of these networks [Golubitsky *et al.*, 2004] is to model systems of differential equations of the form $\underline{\dot{x}} = f(\underline{x})$. Here, the cells model dynamical systems and the edges model couplings between these systems. The values on the cells $0, 1, \ldots, N$ represent the states $x_0, x_1, \ldots, x_N$ of their associated systems, and $\underline{x} = (x_0, x_1, \ldots, x_N)$ is the overall state of the whole system. The connections shown by the edges describe how these values affect each others' rates of change. Several uses of this model are referenced in [Golubitsky & Stewart, 2006]. The general formulation we consider here permits multiple edges, as well as edges from a cell to itself.

Stewart *et al.* [2003], Golubitsky *et al.* [2005] showed that when a coupled cell network representing a system of differential equations has a structure and initial conditions with "local" symmetries given by a kind of relation called a *balanced equivalence relation*, then these symmetries are maintained over time. The converse is also true with suitable technical hypotheses [Stewart *et al.*, 2003, Theorem 6.5]. Thus we may determine striking, possibly quantitative, symmetry properties of the solution of a set of differential equations without finding any "solution" in the usual sense. The relation of equality is trivially balanced, as we shall see. A natural question is whether a given network has any nontrivial balanced equivalence relations. We give an algorithm to answer this question, and show that it runs in polynomial time on total number of cells and edges.

The remainder of this paper is structured as follows.

Sections 2–4 are largely devoted to definitions, introduction of notation, and preliminary results. The notation here gets somewhat involved; this is fairly usual for this field, and happens because we are defining several rather similar concepts, which therefore have similar (but distinct) notation.

In Sec. 3, we give definitions of networks and trees, and "bunching" of trees.

In Sec. 4 we define input trees, homomorphisms on input trees and input tree equivalence, and use these to define balanced equivalence relations on networks. We then describe a partial ordering on these relations. It is known [Stewart, 2007] that the set of balanced equivalence relations on a given locally finite network forms a complete lattice in the sense of Davey and Priestley [1990]. We give a more elementary proof that the set of balanced equivalence relations of an arbitrary network forms a complete lattice. The minimal element of this lattice is equality; we describe the maximal element here, answering in theory the question of existence of nontrivial balanced equivalence relations.

In Sec. 5 we describe an algorithm to determine this maximal balanced relation on a given network. We prove it to be correct, and to execute in polynomial time. We do this first in the case where the network is unitary (that is, all edges have a multiplicity of 1) and then extend to the more general case, taking multiplicity into account.

In this paper, we have adopted a generalized form of "multiplicity" as a real number. The reader with an interest in ODEs with all coefficients equal to 1 can ignore multiplicities (in effect, assuming all multiplicities to be equal to 1) and also ignore the notion of "bunching", and the second algorithm. The reader with an interest in ODEs with natural number coefficients has two options. An arrow in the derived network with multiplicity n can be replaced with n arrows of the same type with multiplicity 1, reducing this case to the previous one (multiple identical arrows are handled transparently by the algorithm), but increasing the running time to be a function of the total multiplicity. Alternatively, consider multiplicities and bunching as described in the second algorithm. For problems with (positive) rational multiplicities it is possible to multiply all arrows by the lowest common multiple of the denominators, and then replace arrows as in the natural number case. However, this leads to a huge problem size, so it is better to consider bunching. Certain problems, such as Markov process networks, can give irrational multiplicities, in which case even this "multiplication and replacement" option is no longer available, forcing the use of the second algorithm. Thus this generalized formalism has practical benefits, as well as being conceptually "tidy". In fact, although only described as a real number in this paper, multiplicities from any abelian group can be handled as described here, and this generalization is potentially useful in connection with such issues as the stability of equilibria.

2. Fundamentals

Here we briefly define some pieces of set theory and relation terminology.

A *multiset* is, informally, a set-like object X such that each item described in the definition of X gives precisely one element of X: although the order of representation of X is ignored, the number of times each distinct element appears is preserved. Effectively, a multiset X is a function from some index set to the "content set" of distinct elements in X, where two multisets A and B are considered equal if there is a bijection ϕ from the index set of A to that of B such that $A\phi = B$, treating A and B as functions. When defining a multiset, we use one of the notations $X = \langle\!| x, x, y, z |\!\rangle$ and $X = \langle\!| f(y) | y \in Y, C(y) |\!\rangle$ for some condition C, as when defining a set. Crucially, when using the second of these notations, there is one element of X for every element of Y for which C is true, even if f is not injective.

Given two equivalence relations $\sim$ and $\approx$ on a set S, $\sim$ is a *refinement* of $\approx$ if $x \sim y \Rightarrow x \approx y$ for $x, y \in S$.

The equivalence relation $\top$ on any set S is defined as: $x \top y \; \forall x, y \in S$.

The term *lattice* is used here as in [Davey & Priestley, 1990], to mean a partially ordered set X in which any two elements $x, y \in X$ have a unique join, denoted $x \vee y$, and meet, denoted $x \wedge y$.

A lattice X is *complete* if every subset $Y \subseteq X$ has a unique greatest lower bound, or *meet*, and a unique least upper bound, or *join*. We denote the meet of Y by $\bigwedge Y$ and its join by $\bigvee Y$. In particular, any finite lattice is complete, and a complete lattice has a maximal and a minimal element.

The dual of Theorem 2.16 from Davey and Priestley [1990] will be useful; we give this dual here:

Theorem 2.1. *Let X be a nonempty partially ordered set. Then the following properties are* *equivalent*:

(1) *X is a complete lattice.*
(2) *$\bigvee Y$ exists for all $Y \subseteq X$.*
(3) *X has a minimal element, and $\bigvee Y$ exists for all $Y \subset X$ with $Y \neq \varnothing$.*

We define a partial order $\leq$ on the equivalence relations on a given set S by $\sim \leq \approx$ if $\sim$ is a refinement of $\approx$. Naturally, we use the symbol $\geq$ for the opposite order. It should be clear that $=$ is the minimal relation on any set S, and $\top$ is the maximal relation. In fact, the set of equivalence relations on S is a complete lattice. The meet $\bigwedge Y$ of a set Y of equivalence relations on S is their intersection when considered as subsets of $S \times S$: that is, $x \bigwedge Y\, y$ if $x \sim y$ for all $\sim \in Y$. The union of a set Y of equivalence relations may not be a transitive relation, but the join $\bigvee Y$ of Y is given by the transitive closure of this union: $x \bigvee Y\, y$ if there is a chain $x = x_0 \sim_1 x_1 \sim_2 \cdots \sim_n x_n = y$ with $x_i \in S$ and $\sim_i \in Y$ for $1 \leq i \leq n$.

3. Networks

We define "networks" in a similar way to that in which coupled cell networks are defined in [Golubitsky *et al.*, 2005]. However, as mentioned in Sec. 1, we permit edges to have a real-valued multiplicity, not just a positive integer multiplicity.

3.1. *Network terminology*

A network $\mathcal{N} = (C, E)$ is a set of cells C and a set of edges E. Each edge is a tuple $e = (c, d, \lambda, i)$, where c and d are cells, called the *tail* and *head* of e respectively, and $\lambda \in \mathbb{R}$ is called the *multiplicity* of e. The number $i \in I$ (for some set I) is called the *identifying mark* of e — it is chosen to allow multiple otherwise equal edges to exist independently in the set E, and has no semantic value. The function $\eta(e)$ gives the head of an edge e, and $\tau(e)$ gives the tail of an edge e. The head and tail of an edge are collectively called its *ends*. The notation $|e|$ denotes the multiplicity of an edge e, so for the edge $e = (c, d, \lambda, i)$, we have $\eta(e) = d$ and $\tau(e) = c$. In particular, and in contrast to [Stewart *et al.*, 2003], we permit the case $c = d$: such edges are called *loops*. Figure 1 shows a diagram representation of a network. Networks where all edges have multiplicity 1 are called *unitary*. See Fig. 2 for an example of a unitary network.

In addition to the sets C, E, the network is equipped with an equivalence relation $\sim_C$ on C

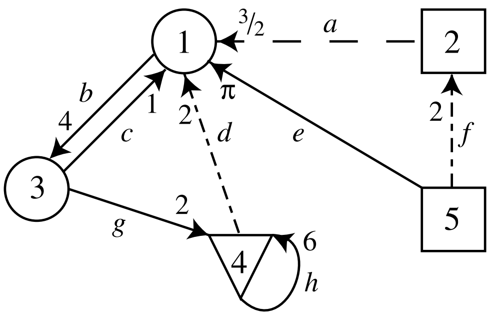


Fig. 1. A (finite, connected) *network* with five cells, $C = \{1, 2, 3, 4, 5\}$ and eight edges, $E = \{a, b, c, d, e, f, g, h\}$. h is a loop. As shown by the shapes of cells and edges, $1 \sim_C 3$, $2 \sim_C 5$; $b \sim_E c \sim_E e \sim_E g \sim_E h$, also $d \sim_E f$.

which assigns each cell c a *cell type* given by its equivalence class, $[c]$, and an equivalence relation $\sim_E$ on E which similarly assigns each edge e an *edge type* $[e]$. In diagrams, such as Fig. 1, cells that are $\sim_C$-equivalent are drawn with the same shape of symbol (circle, square etc.) and edges that are $\sim_E$-equivalent are drawn with the same style of arrow (solid, dashed, dotted etc.). Where we wish to consider both of these relations together, we may use $\sim_{\mathcal{N}}$ to denote $\sim_C \cup \sim_E$. There are at most $|C|$ cell and $|E|$ edge equivalence classes: we consider only networks with a finite number of cell and edge types, even where C or E are infinite sets. Enumerate each of these equivalence class sets, $\{[c] | c \in C\}$ and $\{[e] | e \in E\}$. Denote the number associated with $[c]$ by $[[c]]$ and similarly for $[e]$ and $[[e]]$. A network with only one cell type and one edge type, as in Fig. 2, is called a *homogeneous* network. The triple (c, d, λ) is called the *signature* of $e = (c, d, \lambda, i)$. Two

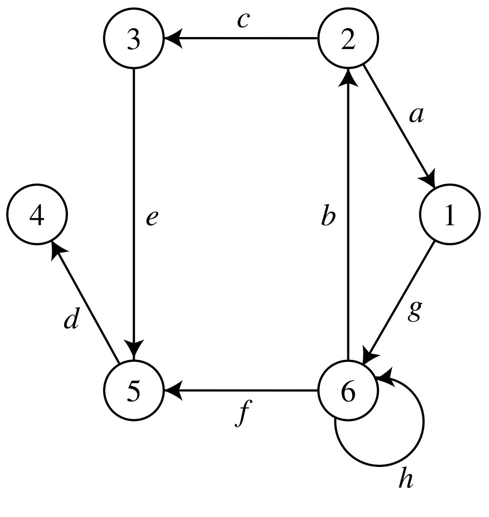


Fig. 2. A *unitary*, *homogeneous* network with six cells and eight edges.

edges e, f with equal signatures are called *identical*, and we write $e \equiv f$.

Networks with a countable or finite number of cells and edges are called *countable* or *finite* networks, respectively. Networks are called *locally countable* or *locally finite* if they have only a countable or finite set of edges meeting any given cell, that is, for all cells $c \in C$, the set $\{e \in E | \eta(e) = c$ or $\tau(e) = c\}$ is countable or finite, respectively. We shall consider only countable, locally finite, networks. (We take the term "countable" to include "finite".)

Given two networks $\mathcal{N} = (C, E), \mathcal{M} = (D, F)$, a *network homomorphism* $\phi : \mathcal{N} \to \mathcal{M}$ is a pair of functions $C \to D$ and $E \to F$, both denoted ϕ, such that:

(1) ϕ preserves the type classes of cells and edges: if $c \sim_C d$ and $e \sim_E f$, then $\phi(c) \sim_D \phi(d)$ and $\phi(e) \sim_F \phi(f)$.
(2) ϕ preserves network structure: if e is an edge in $\mathcal{N}$ with multiplicity λ, head h and tail t, then $\phi(e)$ is an edge in $\mathcal{M}$ with multiplicity λ, head $\phi(h)$ and tail $\phi(t)$.

A *network isomorphism* is a bijective network homomorphism.

Given a network $\mathcal{N}$, let $\sim$ denote a pair of (equivalence) relations, one on the cells of $\mathcal{N}$, the other on its edges. We call this kind of relation pair an (*equivalence*) *relation on* $\mathcal{N}$. For example, $\sim_{\mathcal{N}}$ is an equivalence relation on $\mathcal{N}$.

Given two networks $\mathcal{N}, \mathcal{M}$, let $\sim$ denote a pair of (equivalence) relations, one on the disjoint union of the cell sets of $\mathcal{N}$ and $\mathcal{M}$, the other on the disjoint union of their edge sets. We call this kind of relation pair an (*equivalence*) *relation across* $\mathcal{N}$ *and* $\mathcal{M}$.

Let $\sim$ be a relation across two networks $\mathcal{N} = (C, E)$ and $\mathcal{M} = (D, F)$. We say $\mathcal{N}$ is *isomorphic to* $\mathcal{M}$ *with respect to* $\sim$ if there is an isomorphism $F : \mathcal{N} \overset{\cong}{\to} \mathcal{M}$ which preserves equivalence classes under $\sim$ — that is, $F(x) \sim x$ for cells and edges x in $\mathcal{N}$. We denote this by $\mathcal{N} \cong_\sim \mathcal{M}$.

3.2. *Trees*

Two cells c, d in a network are *connected* if there is a sequence of cells $c = c_0, c_1, \dots, c_n = d$ such that for each i there is an edge with c_i, c_{i+1} as its ends. A *connected network* is a network whose cells are pairwise connected. We define a metric d on the cells of a network. If c, d are connected, then $\mathrm{d}(c, d)$ is the smallest n such that $c = c_0, c_1, \dots, c_n = d$ is a

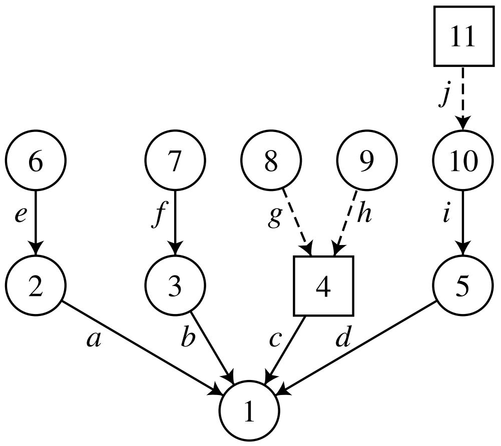


Fig. 3. A *tree* with root 1 and leaves 6, 7, 8, 9, 11. Cells 2, 3, 4, 5 are at generation 1; 6, 7, 8, 9, 10 at generation 2 and 11 is at generation 3; the depth of this tree is 3.

sequence of cells connecting c and d. If c, d are not connected, $\mathrm{d}(c, d) = \infty$.

A *tree* is a connected network which has some cell c_0, called the *root*, such that:

(1) There is no edge with c_0 as its tail.
(2) Given any other cell $c \neq c_0$, there is a unique edge with c as its tail.

A *leaf* of a tree is a cell c such that there is no edge with c as its head. The *generation* of a cell c in a tree is the distance $\mathrm{d}(c_0, c)$. The *depth* of a tree $\mathcal{T}$, $\mathrm{depth}(\mathcal{T})$, is the maximal generation of cells in the tree. If there are cells in $\mathcal{T}$ at generation n for all $n \in \mathbb{N}$, then we define $\mathrm{depth}(\mathcal{T}) = \infty$. See Fig. 3.

Given a tree $\mathcal{T}$ containing some cell c, the *subtree of* $\mathcal{T}$ *rooted at* c, $\mathcal{T}_{(c)}$, is the tree containing all cells d of $\mathcal{T}$ which have a directed path from d to c in $\mathcal{T}$, and all the edges used in such paths. See Fig. 4.

Lemma 3.1. *Any network homomorphism* F *between trees* $\mathcal{T}, \mathcal{U}$ (*including infinite trees*) *must take a cell at the* mth *generation of* $\mathcal{T}$ *to the* mth *generation of* $\mathcal{U}$, *that is*, *tree homomorphisms preserve generation.*

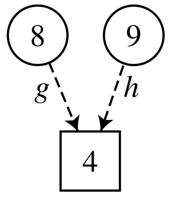


Fig. 4. The *subtree* of Fig. 3 rooted at 4.

Proof. Let $\mathcal{T}$ and $\mathcal{U}$ be trees, and let F be a network (tree) homomorphism between them. The root of $\mathcal{T}$ is the unique cell c_0 which does not appear as the tail of some edge in $\mathcal{T}$, and the root of $\mathcal{U}$ is similarly the cell d_0 which does not appear as a tail in $\mathcal{U}$. So F, being a homomorphism, must map c_0 to d_0. Let c be a cell at the mth generation of $\mathcal{T}$. Since $\mathcal{T}$ is a tree with root c_0, there is a unique (directed) path of length $m = \mathrm{d}(c, c_0)$ from c to c_0 in $\mathcal{T}$. Under F, this maps to a path from the cell $d = F(c)$ to $F(c_0) = d_0$, the root of $\mathcal{U}$. Any shorter path from d to d_0 would connect d to d_0 in two distinct ways — this would give some cell in $\mathcal{U}$ at the tail of two distinct edges. This is impossible since $\mathcal{U}$ is a tree. ■

3.3. *Operations on trees*

The *restriction* of a tree $\mathcal{T}$ to depth n, denoted $\mathcal{T}|_n$, is the tree with cells c from $\mathcal{T}$ where the generation of c is at most n, and all edges (c, d, λ, i) from $\mathcal{T}$ where both c and d are of generation at most n. See Fig. 5. The cell and edge type equivalence relations $\sim_C$ and $\sim_E$ on $\mathcal{T}|_n$ are precisely those on $\mathcal{T}$, restricted to the cells and edges of $\mathcal{T}|_n$. Obviously, the depth of $\mathcal{T}|_n$ is at most n: specifically, it is $\min\{n, \mathrm{depth}(\mathcal{T})\}$, and where the depth of $\mathcal{T}$ is less than or equal to n, we have $\mathcal{T}|_n = \mathcal{T}$.

Lemma 3.2. *For locally finite trees $\mathcal{T}$, the tree $\mathcal{T}|_n$ is finite for all n.*

Proof. The proof is routine, and omitted. ■

Given two trees, $\mathcal{T} = (C, E)$ with a leaf c and $\mathcal{U} = (D, F)$ with root d, the *join* of $\mathcal{U}$ to $\mathcal{T}$ at c is a tree with cell set given by the union of C_* and D_* where $C_* = \{(c', 1) | c' \in C\}$, $D_* = \{(d', 2) | d' \in D \backslash d\}$, and edge set the union of E_* and F_*, where

$$E_* = \{((c_1, 1), (c_2, 1), \lambda) | (c_1, c_2, \lambda) \in E\}$$

and

$$F_* = \{((d_1, 2), (d_2, 2), \lambda) | (d_1, d_2, \lambda) \in F \text{ and } d_1 \neq d\} \cup \{((c, 1), (d_2, 2), \lambda) | (d, d_2, \lambda) \in F\}$$

See Fig. 6 for an example. Informally, we ensure that C and D are disjoint, and then identify c with d, while keeping all edges the same. Cell and edge equivalences are, by default, defined as the union of those of $\mathcal{T}$ and $\mathcal{U}$, with the additional condition that all cells which are equivalent to d in $\mathcal{U}$ become equivalent to all cells which are equivalent to c in $\mathcal{T}$. In this case, no cell in $\mathcal{T}$ which is not equivalent to c becomes equivalent to a cell in $\mathcal{U}$, and no cell in $\mathcal{U}$ which is not equivalent to d becomes equivalent to a cell in $\mathcal{T}$. However, as we shall see later, there are cases when the cell and edge equivalence relations on $\mathcal{T}$ and $\mathcal{U}$ are in fact restrictions of a single pair of equivalence relations $\sim_C, \sim_E$ on the unions of the cells and edges of these trees. In that case, we may take the restrictions of the relations $\sim_C, \sim_E$ to the join of $\mathcal{T}$ and $\mathcal{U}$ as the equivalences on this new tree, again defining $(c', 1) \sim_C (d', 2)$ whenever $c' \sim_C c$ and $d' \sim_C d$.

Let two pairs of trees, $(\mathcal{T}_i, \mathcal{U}_i)$ for $i = 1, 2$ be given. Let $(\mathcal{T}_1, \mathcal{U}_1)$ satisfy the conditions for $(\mathcal{T}, \mathcal{U})$ in the definition of the join of trees, with a leaf c_1 of $\mathcal{T}_1$ and root d_1 of $\mathcal{U}_1$. Let $f : \mathcal{T}_1 \to \mathcal{T}_2$ and $g : \mathcal{U}_1 \to \mathcal{U}_2$ be homomorphisms of trees, let $c_2 = f(c_1)$ and $d_2 = g(d_1)$. By Lemma 3.1, d_2 is the root of $\mathcal{U}_2$. Suppose c_2 is a leaf in $\mathcal{T}_2$ — for example, if f is an isomorphism. Then the join of $\mathcal{T}_2$ to $\mathcal{U}_2$ at c_2 is well-defined. We define the *join* of f and g at c_1 as a function on the join of $\mathcal{T}_1$ and $\mathcal{U}_1$ which acts as f on $\mathcal{T}_1$ and as g on $\mathcal{U}_1$. The only point in the intersection of these trees is the leaf c_1, identified with d_1. f maps this cell to c_2, and g maps it to d_2: these cells are identified in the join of $\mathcal{T}_2$ and $\mathcal{U}_2$. Thus this join is a well-defined homomorphism — further, if both

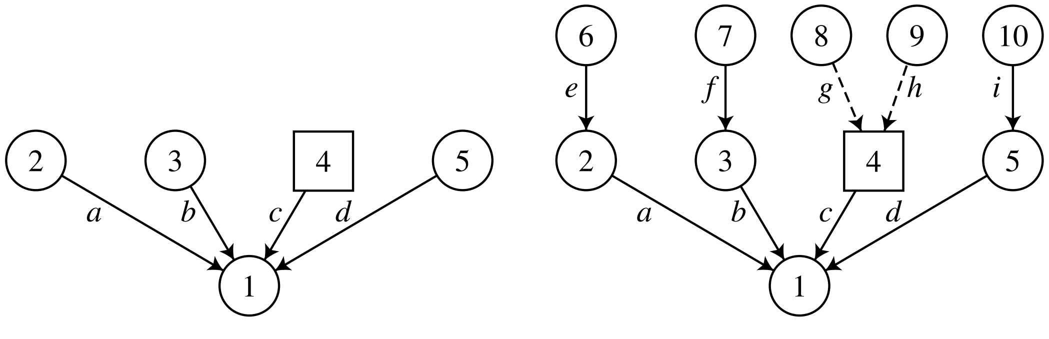


Fig. 5. The restriction of Fig. 3 to depths 1 (left) and 2 (right).

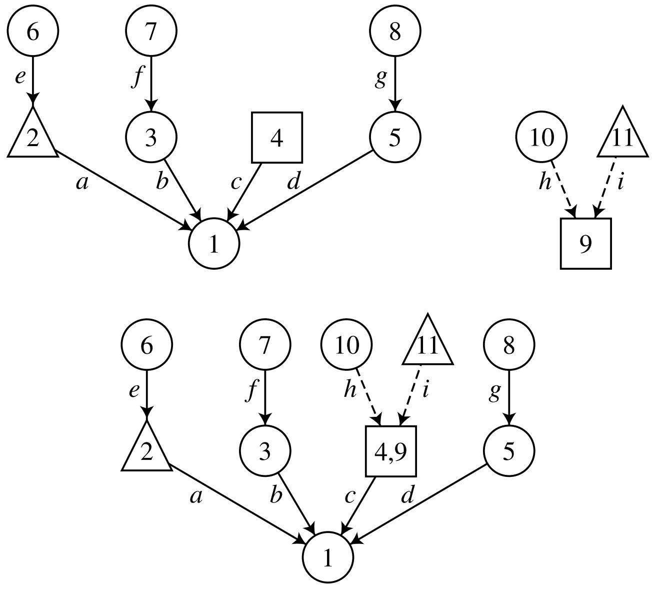


Fig. 6. Trees $\mathcal{T}$ (top left) and $\mathcal{U}$ (top right) and the *join* of $\mathcal{T}$ to $\mathcal{U}$ at 4. Note that some cells in $\mathcal{T}$ are equivalent to some in $\mathcal{U}$, as shown by shapes (for example, $2 \sim_C 11$), and in particular, $4 \sim_C 9$.

f and g are isomorphisms, then this join is also an isomorphism.

3.4. *Sequences of trees*

We now consider sequences of trees, and their limits.

Given a sequence of trees $(\mathcal{T}_i)$ such that there is a sequence of isomorphisms $f_i : \mathcal{T}_i|_{n_i} \overset{\cong}{\rightarrow} \mathcal{T}_{i+1}|_{n_i}$ for some sequence $(n_i) \to \infty$, we call $(\mathcal{T}_i)$ a *convergent* sequence. We call (f_i) the *convergence isomorphism* of $(\mathcal{T}_i)$.

Given a convergent sequence of trees $(\mathcal{T}_i)$, we construct $T_\infty = \lim_{i\to\infty} T_i$, the *direct limit* of $(\mathcal{T}_i)$, as follows. Let $\mathcal{T}_\infty \supseteq \mathcal{T}_i$ for all i, by defining

$$\mathcal{T}_\infty = \left(\frac{\dot{\bigcup\limits_{i\in\mathbb{N}}} C_i}{\bigcup\limits_{i\in\mathbb{N}} f_i}, \frac{\dot{\bigcup\limits_{i\in\mathbb{N}}} E_i}{\bigcup\limits_{i\in\mathbb{N}} f_i} \right)$$

That is, after taking the disjoint unions of cell and edge sets, we identify all cells and edges mapped to each other by any of the individual isomorphisms f_i from the convergence isomorphism. The disjoint unions — of the cells, for example — can be formed by passing from the set C_i to the set $C_i \times \{i\}$, and then the quotient can be formed by an equivalence relation $\sim$ in which $(c, i) \sim (f_i(c), i+1)$ for all i. We generally refer to the cell (c, i) and the edge (e, i) just as c and e.

Let $(\mathcal{T}_i), (\mathcal{U}_i)$ be two convergent sequences of trees with convergence isomorphisms $(f_i), (g_i)$ respectively. Given a sequence of functions $(F_i : \mathcal{T}_i \to \mathcal{U}_i)$ such that, for all cells c and edges e in $\mathcal{T}_i$, there is some natural number n such that:

$$\left.\begin{aligned} F_{i+1}(f_i(c)) &= g_i(F_i(c)) \\ F_{i+1}(f_i(e)) &= g_i(F_i(e)) \end{aligned}\right\} \quad \text{for } i \geq n \qquad (1)$$

the *direct limit* of (F_i) is a function $\lim_{n\to\infty} F_n = F_\infty : \mathcal{T}_\infty \to \mathcal{U}_\infty$ which takes a cell c in $\mathcal{T}_\infty$ identified with a cell c_i in $\mathcal{T}_i$ (for sufficiently high i) to the cell d in $\mathcal{U}_\infty$ identified with $d_i = F_i(c_i)$ in $\mathcal{U}_i$. Condition 1 ensures that this is well defined.

Lemma 3.3. *If $F : \mathcal{T} \to \mathcal{U}$ is a homomorphism of infinite trees $\mathcal{T}, \mathcal{U}$, then each restriction $F|_{\mathcal{T}|_n} : \mathcal{T}|_n \to \mathcal{U}|_n$ is also a homomorphism. If F is an isomorphism, so is each $F|_{\mathcal{T}|_n}$. Further, if $\sim$ is an equivalence relation across $\mathcal{T}$ and $\mathcal{U}$ such that F preserves $\sim$, then each $F|_{\mathcal{T}|_n}$ preserves the restriction of $\sim$ to $\mathcal{T}|_n$, $\mathcal{U}|_n$.*

Proof. Tree homomorphisms preserve generation, so the restriction $F|_{\mathcal{T}|_n}$ is a well-defined function to $\mathcal{U}|_n$. Since F, as a homomorphism, takes the head and tail of an edge in $\mathcal{T}$ to the head and tail of its image in $\mathcal{U}$, the same property must hold in the restriction of F, since the ends of any edge e in a restriction $\mathcal{T}|_n$ of $\mathcal{T}$ must be the same as the ends of e in $\mathcal{T}$ itself, and thus must be taken to the ends of $F(e)$ in $\mathcal{U}$ which are the ends of $F(e)$ in $\mathcal{U}|_n$.

The preservation of generation also ensures that if $F : \mathcal{T} \to \mathcal{U}$ is surjective, then so is $F|_{\mathcal{T}|_n} : \mathcal{T}|_n \to \mathcal{U}|_n$, since a cell (or edge) of $\mathcal{U}$ at generation $m \leq n$ can only be mapped to by F from a cell (or edge) of $\mathcal{T}$ at generation $m \leq n$, which is therefore also a cell (or edge) of $\mathcal{T}|_n$. Any restriction of an injective function is injective, so if F is a bijection, $F|_{\mathcal{T}|_n}$ is also. ■

The converse to this lemma also holds.

Lemma 3.4. *If $(F_i : \mathcal{T}_i \to \mathcal{U}_i)$ is a sequence of tree homomorphisms for convergent sequences of trees $(\mathcal{T}_i)$, $(\mathcal{U}_i)$ as above, then F_∞ is also a homomorphism, $F_\infty : \mathcal{T}_\infty \to \mathcal{U}_\infty$. If the F_i are isomorphisms, then so is F_∞.*

If $\sim$ is an equivalence relation across $\mathcal{T}_\infty$ and $\mathcal{U}_\infty$ (and, by restriction, across all trees $\mathcal{T}_i, \mathcal{U}_i$) such that F_i respects $\sim$ for all i, then so does F_∞.

Proof. Let $(F_i : \mathcal{T}_i \to \mathcal{U}_i)$ be a sequence of tree homomorphisms for convergent sequences of trees

$(\mathcal{T}_i), (\mathcal{U}_i)$. Let c_i in $\mathcal{T}_i, \mathcal{U}_i$ denote the cells identified with a cell c of $\mathcal{T}_\infty, \mathcal{U}_\infty$ for $i \geq n_c \in \mathbb{N}$. Similarly, let e_i denote the edges identified with an edge e, and n_e be the minimal i such that $\mathcal{T}_i$ or $\mathcal{U}_i$ contains such an edge.

Consider a specific edge e in $\mathcal{T}$ with head $\eta(e) = c$ and tail $\tau(e) = c'$. Let the edge $f = F_\infty(e)$ from $\mathcal{U}_\infty$, and cells $d = F_\infty(c)$ and $d' = F_\infty(c')$.

Let $i \geq n_e$, so that e_i, c_i, c'_i all exist in $\mathcal{T}_i$. By application of the convergence isomorphism of $(\mathcal{T}_i)$, e_i has head c_i and tail c'_i. Since F_i is well defined, $F_i(e_i)$ must be an edge in $\mathcal{U}_i$, and $F_i(c_i), F_i(c'_i)$ must be cells in $\mathcal{U}_i$. By Condition 1, F_i must take e_i, corresponding to e, to f_i, corresponding to f. Similarly, $F_i(c_i) = d_i$ and $F_i(c'_i) = d'_i$. Since F_i is a homomorphism, f_i must have head d_i and tail d'_i. Thus f must have head d and tail d', and F_∞ is a homomorphism.

If each F_i is bijective, they have well-defined inverses — the direct limit of these gives a well-defined inverse to F_∞, which is therefore also bijective. Thus if each F_i is a tree isomorphism, F_∞ is also an isomorphism.

Suppose $\sim$ is an equivalence relation across $\mathcal{T}_\infty$ and $\mathcal{U}_\infty$, and define $\sim$ across all $\mathcal{T}_i, \mathcal{U}_i$ by $c \sim c_i$, $e \sim e_i$ for all cells c and edges e of $\mathcal{T}_\infty, \mathcal{U}_\infty$. Then if F_i preserves $\sim$ for some $i \geq n_c$, we have $c \sim c_i \sim F_i(c_i) = d_i \sim d = F_\infty(c)$. So F_∞ also preserves $\sim$. ■

Lemma 3.5. *For any tree $\mathcal{T}$, $\mathcal{T} \cong \lim_{n\to\infty} \mathcal{T}|_n$.*

Proof. From the remark in the definition of the restriction of a tree, this holds trivially for all finite trees $\mathcal{T}$.

Let $\mathcal{T}$ be an infinite tree. Clearly, $\mathcal{T} = \lim_{n\to\infty} \mathcal{T}$. Take the two sequences of trees $(\mathcal{T}_i), (\mathcal{U}_i)$ given by $\mathcal{T}_i = \mathcal{T}|_i$, $\mathcal{U}_i = \mathcal{T}$, where the convergence isomorphism of $(\mathcal{T}_i)$ is given by the inclusion maps, and that of $\mathcal{U}_i$ is the identity map at each i. Then there is a natural, injective, homomorphism $F_i : \mathcal{T}_i \to \mathcal{U}_i$. This function has limit $F_\infty : \lim_{i\to\infty} \mathcal{T}|_i \to \mathcal{T}$, which is a homomorphism of trees, by Lemma 3.4. We show this F_∞ to be a bijection. Let $\mathcal{T}_\infty = \lim_{i\to\infty} \mathcal{T}_i = \lim_{i\to\infty} \mathcal{T}|_i$. Take some cell c in $\mathcal{T}$, then it is mapped to by the cell c itself wherever it appears in $\mathcal{T}|_i$. It appears in all such trees where $i \geq n_c$, so c is the image of at least one cell in $\mathcal{T}_\infty$, containing the cell c from $\mathcal{T}_{n_c}$, say. So F_∞ is surjective. But all these cells c from $\mathcal{T}_i$ mapping to c in $\mathcal{T}$ are identified by the inclusion map, which is the convergence isomorphism of $(\mathcal{T}_i)$. Hence c has a unique preimage in $\mathcal{T}_\infty$, and is an injection. Thus $F_\infty : \mathcal{T} \overset{\cong}{\to} \lim_{n\to\infty} \mathcal{T}|_n$, as required. ■

3.5. *Bunching*

In systems modeled by the networks we are defining here, any set of arrows of the same type between the same cells is usually considered to be equivalent to any other set with the same total multiplicity — in particular, equivalent to a single arrow with appropriate multiplicity. Also, identical cells which have identical inputs will behave identically, and so they may be considered as one cell — effectively, any output arrows from one of the identically-behaving cells can be transferred to the other without affecting the behavior of the network. Transferring all such arrows, the former cell then has no effect on the network, and can be removed. To inspect the value of the removed cell, we may take the value of the equivalent cell that is left in the network. We now formalize this process on trees, which is where we will later use it.

Given a tree $\mathcal{T} = (C, E)$ and an equivalence relation $\sim$ on $\mathcal{T}$, we define a tree called the *bunching of $\mathcal{T}$ with respect to* $\sim$, denoted $\mathcal{T}/\!\!/\!\sim$. Firstly, assume $\mathcal{T}$ is finite. Define a relation $\dot\sim$ on E by $e \,\dot\sim\, f$ where either $e = f$, or all of the following: $e \sim f$, $c = \tau(e) \sim \tau(f) = d$ and the subtrees rooted at c, d are isomorphic with respect to $\sim$ after being bunched by $\sim$ — that is, $\mathcal{T}_{(c)}/\!\!/\!\sim \,\cong_\sim\, \mathcal{T}_{(d)}/\!\!/\!\sim$. Clearly, this is an inductive definition requiring descending induction. Now identify edges e, f and cells $\tau(e), \tau(f)$ where $e \,\dot\sim\, f$, provided $\eta(e)$ is identified with $\eta(f)$. See Fig. 7. Formally, define a third relation, $\hat\sim$, between cells and edges:

$$c \,\hat\sim\, d \Leftrightarrow c = d \quad \text{or} \quad \begin{cases} c = \tau(e) \\ d = \tau(f) \\ e \,\hat\sim\, f \end{cases}$$

$$e \,\hat\sim\, f \Leftrightarrow e \,\dot\sim\, f \quad \text{and} \quad \eta(e) \,\hat\sim\, \eta(f)$$

This uses ascending induction. Hence this forms a definition only if $\mathcal{T}$ is finite, due to the descending induction step. Now the bunching of $\mathcal{T}$ with respect to $\sim$ is a tree formed from the partition of C and E given by this $\hat\sim$, that is, $\mathcal{T}/\!\!/\!\sim$ is defined to be $\mathcal{T}/\hat\sim$, in the usual sense of quotient by an equivalence relation. We call the parts of C and E *cell* and *edge bunches*. Given an edge bunch, its *head* and *tail* are the cell bunches containing, respectively, the heads and tails of its constituent edges: each of these

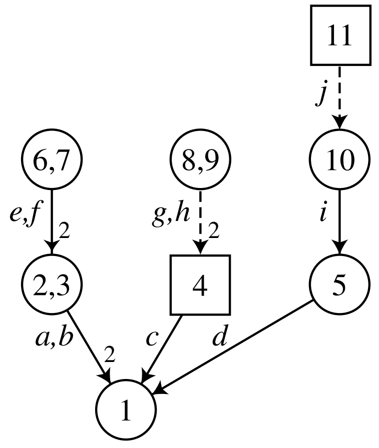


Fig. 7. The network shown in Fig. 3, bunched by the cell and edge equivalence relations $\sim_C$ and $\sim_E$.

is a single well-defined cell bunch, since edges are only identified $e \mathrel{\widehat{\sim}} f$ where $\eta(e) \mathrel{\widehat{\sim}} \eta(f)$, and similarly, if edges $e \mathrel{\widehat{\sim}} f$, their tails $\tau(e) \mathrel{\widehat{\sim}} \tau(f)$. Provided $\sim$ refines $\sim_C$ and $\sim_E$, cell and edge bunches have well-defined types derived from the types of their members. The cardinality of each edge bunch is exactly equal to the cardinality of its tail cell bunch, but instead of cardinality, we interest ourselves with multiplicity of edge bunches. The *multiplicity* of an edge bunch is defined as the sum of the multiplicities of its constituent edges. We can now consider $\mathcal{T}/\!\!/\!\sim$ to be a tree in its own right with its "cells" and "edges" being cell and edge bunches. Figure 7 shows a bunched tree displayed as a tree in this way. Note that passing from a cell in a tree $\mathcal{T}$ to the cell bunch which contains it in the bunched tree $\mathcal{T}/\!\!/\!\sim$ preserves generation. Also, it is clear that this process relies only on the structure (up to isomorphism) of $\mathcal{T}$ and the equivalence relation $\sim$. Hence, given two trees $\mathcal{T},\mathcal{U}$ and a relation $\sim$ across them such that $\mathcal{T} \cong_\sim \mathcal{U}$, we have $\mathcal{T}/\!\!/\!\sim \,\cong_\sim \mathcal{U}/\!\!/\!\sim$.

For general locally finite trees $\mathcal{T}$, including infinite trees, recall $\mathcal{T} = \lim_{n\to\infty} \mathcal{T}|_n$, from Lemma 3.5. Define $\mathcal{T}/\!\!/\!\sim \,= \lim_{n\to\infty}(\mathcal{T}|_n/\!\!/\!\sim)$.

Lemma 3.6. *The limit $\mathcal{T}/\!\!/\sim$ is well defined.*

Proof. We must show $(\mathcal{T}|_n/\!\!/\!\sim)$ to be a convergent sequence of trees.

Let c, d be two cells at the same generation g of $\mathcal{T}$ which are not identified in all $\mathcal{T}|_n/\!\!/\!\sim$. Then there is some $n_{c,d}$ such that c and d are in distinct cell bunches of $\mathcal{T}|_m/\!\!/\!\sim$ precisely when $m \geq n_{c,d}$.

By Lemma 4.1, there is a finite number of cells at each generation g of $\mathcal{T}$, and so there is a finite number of pairs of such cells (c, d), and hence a finite number of $n_{c,d}$ defined as above, for each generation g. Let $n_g = \max\{n_{c,d}|c, d \text{ are at generation } g\}$. Then for $m \geq n_g$, all cells at generation g (and below) are bunched in the same way by $\mathcal{T}|_m/\!\!/\!\sim$. Hence the trees $(\mathcal{T}|_m/\!\!/\!\sim)|_g$ are all isomorphic for $m \geq n_g$.

For a given $n \in \mathbb{N}$, consider $g_n = \max\{g|n_g < n\}$. This is the maximal generation g such that the "final" cell bunches of all cells at generation g are determined by bunching the tree $\mathcal{T}|_m$, and in particular, $(\mathcal{T}|_m/\!\!/\!\sim)|_{g_n} \cong (\mathcal{T}|_{m+1}/\!\!/\!\sim)|_{g_n}$. Every generation's bunching must be determined in this way at some point (since n_g is finite for all $g \in \mathbb{N}$), so as $n \to \infty$, $g_n \to \infty$. This gives the required convergence isomorphism for $(\mathcal{T}|_n/\!\!/\!\sim)$. ■

Let $\mathcal{T},\mathcal{U}$ be two trees, and let $\sim$ be an equivalence relation across $\mathcal{T}$ and $\mathcal{U}$. We say $\mathcal{T}$ and $\mathcal{U}$ are *equivalent with respect to* $\sim$ when they are isomorphic with respect to $\sim$ when bunched by $\sim$, that is, $\mathcal{T}/\!\!/\!\sim \,\cong_\sim \mathcal{U}/\!\!/\!\sim$.

Lemma 3.7. *Let $\mathcal{T}$ be a tree and $\sim$ an equivalence relation on $\mathcal{T}$. Let c be a cell of $\mathcal{T}$, corresponding to a cell bunch $\mathbf{c}_\sim$ in $\mathcal{T}/\!\!/\!\sim$. Then $\mathcal{T}_{(c)}/\!\!/\!\sim \,\cong_\sim (\mathcal{T}/\!\!/\!\sim)_{(\mathbf{c}_\sim)}$.*

Proof. Let $\mathcal{T}$ be a tree and $\sim$ an equivalence relation on $\mathcal{T}$. Let c be a cell of $\mathcal{T}$, corresponding to a cell bunch $\mathbf{c}_\sim$ in $\mathcal{T}/\!\!/\!\sim$. All bunchings $\mathcal{T}_{(c')}/\!\!/\sim$ of subtrees $\mathcal{T}_{(c')}$ rooted at $c' \in \mathbf{c}_\sim$ must be isomorphic to each other with respect to $\sim$, by the definition of bunching. The bunched subtree $(\mathcal{T}/\!\!/\!\sim)_{(\mathbf{c}_\sim)}$ is precisely the identification of these trees. ■

Proposition 3.8. *If $\mathcal{T}$ and $\mathcal{U}$ are two trees, and $\sim \,\leq\, \approx$ are equivalence relations across $\mathcal{T}$ and $\mathcal{U}$ such that $\mathcal{T}$ is equivalent to $\mathcal{U}$ with respect to $\sim$, then $\mathcal{T}$ is also equivalent to $\mathcal{U}$ with respect to $\approx$. That is, $\mathcal{T}/\!\!/\!\sim \,\cong_\sim \mathcal{U}/\!\!/\!\sim \,\Rightarrow \mathcal{T}/\!\!/\!\approx \,\cong_\approx \mathcal{U}/\!\!/\!\approx$.*

Proof. The proof is quite straightforward, although somewhat lengthy.

Let $\mathcal{T},\mathcal{U},\sim,\approx$ be as described in the hypotheses above. We only need to show the conclusion holds for finite trees, since if $F : \mathcal{T}/\!\!/\!\sim \,\xrightarrow{\cong_\sim} \mathcal{U}/\!\!/\!\sim$, the restriction $F_n = F|_{\mathcal{T}/\!\!/\sim|_n}$ is an isomorphism which respects $\sim$, $F_n : \mathcal{T}/\!\!/\!\sim \,\xrightarrow{\cong_\sim} \mathcal{U}/\!\!/\!\sim$ for each $n \in \mathbb{N}$ (by Lemma 3.3). Each of these tree restrictions is a finite tree, so provided the proposition holds for all finite trees there is a sequence of functions $(G_n : \mathcal{T}/\!\!/\!\approx|_n \xrightarrow{\cong_\approx} \mathcal{U}/\!\!/\!\approx|_n)$. By Lemma 3.4 there is a tree homomorphism $G_\infty = \lim_{n\to\infty} G_n$, such that $G_\infty : \mathcal{T}/\!\!/\approx \xrightarrow{\cong_\approx} \mathcal{U}/\!\!/\approx$ is a tree isomorphism with respect to $\approx$.

It only remains to show:

$$\mathcal{T}/\!\!/\sim \,\cong_\sim \mathcal{U}/\!\!/\sim \;\Rightarrow\; \mathcal{T}/\!\!/\approx \,\cong_\approx \mathcal{U}/\!\!/\approx \qquad (2)$$

for all finite trees, which we do by induction on the greater of the depths of $\mathcal{T}$,$\mathcal{U}$.

Clearly, if depth($\mathcal{T}$) = depth($\mathcal{U}$) = 0, then $\mathcal{T}$ and $\mathcal{U}$ are single cells: call these t and u, and passing to the bunched trees $\mathcal{T}/\!\!/\sim$,$\mathcal{U}/\!\!/\sim$ has no effect: that is, $\mathcal{T} \cong_\sim \mathcal{T}/\!\!/\sim$ and $\mathcal{U} \cong_\sim \mathcal{U}/\!\!/\sim$, and similarly for $\approx$. If $\mathcal{T} \cong_\sim \mathcal{T}/\!\!/\sim \,\cong_\sim \mathcal{U}/\!\!/\sim \,\cong_\sim \mathcal{U}$ then $t \sim u$ and so $t \approx u$. Thus $\mathcal{T}/\!\!/\approx \,\cong_\approx \mathcal{T} \cong_\approx \mathcal{U} \cong_\approx \mathcal{U}/\!\!/\approx$. So Hypothesis 2 holds for trees $\mathcal{T}$,$\mathcal{U}$ of depth 0.

Now suppose the hypothesis 2 holds for all trees $\mathcal{T}$,$\mathcal{U}$ of depth at most $n-1$, and then let $\mathcal{T}$,$\mathcal{U}$ be some given trees of depth n such that $\mathcal{T}/\!\!/\sim \,\cong_\sim \mathcal{U}/\!\!/\sim$. We show $\mathcal{T}/\!\!/\approx \,\cong_\approx \mathcal{U}/\!\!/\approx$, giving 2 for trees of depth n.

For a cell c in $\mathcal{T}$ or $\mathcal{U}$, let $\mathbf{c}_\sim$ denote the cell bunch of $\mathcal{T}/\!\!/\sim$ or $\mathcal{U}/\!\!/\sim$ containing c. Similarly, define $\mathbf{c}_\approx$ in $\mathcal{T}/\!\!/\approx$, and $\mathbf{e}_\sim$,$\mathbf{e}_\approx$ for edges e.

We now show by a second (internal) induction that for trees $\mathcal{T}$,$\mathcal{U}$ of depth n and a relation $\sim$, as in the main inductive hypothesis 2,

$$\mathcal{T}/\!\!/\approx \,\cong_\approx (\mathcal{T}/\!\!/\sim)/\!\!/\approx \qquad (3)$$

and similarly for $\mathcal{U}$. Clearly this is true for trees of depth 0. For the inductive step, assume $m \le n$ is given such that this isomorphism 3 holds for all trees of depth at most $m-1$, and let $\mathcal{T}$ be a tree of depth m.

Now let c, d be cells at generation 1 in $\mathcal{T}$, at the tails of edges e and f, respectively. We show that $\mathbf{c}_\sim = \mathbf{d}_\sim \Rightarrow \mathbf{c}_\approx = \mathbf{d}_\approx$. Suppose that $\mathbf{c}_\sim = \mathbf{d}_\sim$. Then $\mathcal{T}_{(c)}/\!\!/\sim \,\cong_\sim \mathcal{T}_{(d)}/\!\!/\sim$. These subtrees $\mathcal{T}_{(c)}$ and $\mathcal{T}_{(d)}$, rooted at c and d respectively, are of depth at most $n-1$. So by the main inductive hypothesis 2, $\mathcal{T}_{(c)}/\!\!/\approx \,\cong_\approx \mathcal{T}_{(d)}/\!\!/\approx$ (in particular, $c \approx d$). Also $e \sim f$, so $e \approx f$, and both e and f have their heads at the root of $\mathcal{T}$, so e and f are compatible edges for bunching by $\approx$. Thus $\mathbf{c}_\approx = \mathbf{d}_\approx$, as required.

This means that the cell bunches in $\mathcal{T}/\!\!/\approx$ are formed from unions of cell bunches in $\mathcal{T}/\!\!/\sim$. Recall that bunching preserves equivalence class, $c \sim \mathbf{c}_\sim$. So $c \approx \mathbf{c}_\sim$. From Lemma 3.7, $(\mathcal{T}/\!\!/\sim)_{(\mathbf{c}_\sim)} \cong_\sim \mathcal{T}_{(c)}/\!\!/\sim$. Since c is not the root of $\mathcal{T}$, depth($\mathcal{T}_{(c)}$) < depth($\mathcal{T}$), and Hypothesis 2 gives $(\mathcal{T}/\!\!/\sim)_{(\mathbf{c}_\sim)}/\!\!/\approx \,\cong_\approx (\mathcal{T}_{(c)}/\!\!/\sim)/\!\!/\approx$. Also, by Hypothesis 3, $(\mathcal{T}_{(c)}/\!\!/\sim)/\!\!/\approx \,\cong_\approx \mathcal{T}_{(c)}/\!\!/\approx$. So $(\mathcal{T}/\!\!/\sim)_{(\mathbf{c}_\sim)}/\!\!/\approx \,\cong_\approx (\mathcal{T}_{(c)}/\!\!/\sim)/\!\!/\approx \,\cong_\approx \mathcal{T}_{(c)}/\!\!/\approx$. Hence the cell bunches in $\mathcal{T}/\!\!/\approx$ are precisely the unions of cell bunches $\mathbf{c}_\sim$ and $\mathbf{d}_\sim$ from $\mathcal{T}/\!\!/\sim$ where $\mathbf{c}_\sim \approx \mathbf{d}_\sim$ and $\mathcal{T}_{(\mathbf{c}_\sim)} \cong_\approx \mathcal{T}_{(\mathbf{d}_\sim)}$, provided $\mathbf{c}_\sim$ and $\mathbf{d}_\sim$ are tails of some edges $\mathbf{e}_\sim, \mathbf{f}_\sim$ such that $\mathbf{e}_\sim \approx \mathbf{f}_\sim$, and the heads of $\mathbf{e}_\sim, \mathbf{f}_\sim$ are themselves identified. These are precisely the conditions for bunching $\mathbf{c}_\sim$ with $\mathbf{d}_\sim$ (and $\mathbf{e}_\sim$ with $\mathbf{f}_\sim$) when forming the tree $(\mathcal{T}/\!\!/\sim)/\!\!/\approx$. Hence $\mathcal{T}/\!\!/\approx \,\cong_\approx (\mathcal{T}/\!\!/\sim)/\!\!/\approx$ (clearly any isomorphism preserving $\sim$ must preserve $\approx$). This is equally true for $\mathcal{U}$.

Still assuming $\mathcal{T}/\!\!/\sim \,\cong_\sim \mathcal{U}/\!\!/\sim$, we have $\mathcal{T}/\!\!/\sim \,\cong_\approx \mathcal{U}/\!\!/\sim$, so by the remark after the definition of bunching, $(\mathcal{T}/\!\!/\sim)/\!\!/\approx \,\cong_\approx (\mathcal{U}/\!\!/\sim)/\!\!/\approx$.

Thus we have:

$$\mathcal{T}/\!\!/\approx \,\cong_\approx (\mathcal{T}/\!\!/\sim)/\!\!/\approx \,\cong_\approx (\mathcal{U}/\!\!/\sim)/\!\!/\approx \,\cong_\approx \mathcal{U}/\!\!/\approx$$

and in particular, $\mathcal{T}/\!\!/\approx \,\cong_\approx \mathcal{U}/\!\!/\approx$, as required.

The proof follows by induction. ■

3.6. *Operations on bunched trees*

Lemma 3.9. *If $\sim$ is an equivalence relation on a tree $\mathcal{T}$, then $(\mathcal{T}/\!\!/\sim)|_n = \mathcal{T}|_n/\!\!/\sim$.*

Proof. Let $\mathcal{T}$ be a tree, and $\sim$ an equivalence relation on $\mathcal{T}$. A cell bunch $\mathbf{c}$ in $\mathcal{T}/\!\!/\sim$ appears in $(\mathcal{T}/\!\!/\sim)|_n$ precisely when the generation of $\mathbf{c}$ is at most n, so the generation of each $c \in \mathbf{c}$ is also at most n. Thus all such c appear in $\mathcal{T}|_n$, and so the cell bunch $\mathbf{c}$ appears in $\mathcal{T}|_n/\!\!/\sim$. Similarly for edge bunches, an edge bunch $\mathbf{e}$ in $\mathcal{T}/\!\!/\sim$ appears in $(\mathcal{T}/\!\!/\sim)|_n$ precisely when the generation of its tail is at most n (since the generation of its head is less than that of its tail). This is the case when the tails of the constituent edges of $\mathbf{e}$ have generation at most n, thus these edges are in $\mathcal{T}|_n$. Hence $\mathbf{e}$ is an edge of $(\mathcal{T}/\!\!/\sim)|_n$. ■

Lemma 3.10. *Let $\mathcal{T}$, $\mathcal{U}$ be two trees, with some equivalence relation $\sim$ across their cell and edge sets. Let $\mathbf{c}$ be a cell bunch which is a leaf in $\mathcal{T}/\!\!/\sim$ — that is, all $c \in \mathbf{c}$ are leaves. Let $\mathcal{J}$ be the join of $\mathcal{U}$ to $\mathcal{T}$ at each $c \in \mathbf{c}$. Then $\mathcal{J}/\!\!/\sim$ is (isomorphic to) the join of $\mathcal{U}/\!\!/\sim$ to $\mathcal{T}/\!\!/\sim$ at $\mathbf{c}$, considering $\mathcal{T}/\!\!/\sim$ as a tree in its own right, as usual.*

Proof. Since the subtrees rooted at $c \in \mathbf{c}$ are all isomorphic to $\mathcal{U}$, they are all isomorphic, with the same pattern of $\sim$ equivalence classes, so they are all isomorphic when bunched by $\sim$. Thus the cells $c \in \mathbf{c}$ can all be bunched together by $\sim$ in the join $\mathcal{J}$ of $\mathcal{T}$ and $\mathcal{U}$. Since nothing else about $\mathcal{T}$ has changed in $\mathcal{J}$, the part of $\mathcal{J}$ which corresponds to $\mathcal{T}$ is bunched exactly as $\mathcal{T}/\!\!/\!\sim$. Also, the subtree rooted at $\mathbf{c}$ in $\mathcal{J}/\!\!/\!\sim$ is precisely the bunched $\mathcal{U}$. ■

4. Input Trees and Balanced Equivalence Relations

We now define "balanced equivalence relations" on a network $\mathcal{N}$ in terms of certain trees, derived from the structure of $\mathcal{N}$, which we call "input trees". We then (Theorem 4.6) prove that the partially ordered set of these relations forms a lattice, and (Theorem 4.7) characterize the maximal balanced equivalence relation in terms of the "infinite input tree".

4.1. *Input sets and trees*

The *input edges* of a cell c in a network $\mathcal{N} = (C, E)$ are the edges of the form (d, c, λ, i). The *input set* of c is a network $I(c) = (C_c, E_c)$ with cell set

$$C_c = \{c\} \cup \{d_e | e = (d, c, \lambda, i) \text{ is an input edge of } c\}$$

(d_e represents the pair (d, e) when used in this context, for clarity) and edge set

$$E_c = \{(d_e, c, \lambda, i) | e = (d, c, \lambda, i)\}$$

This $I(c)$ is clearly a tree of depth 1, with root c. Cell and edge types are preserved: we extend $\sim_C$ and $\sim_E$ such that $d_e \sim_C d$ and $(d_e, c, \lambda, i) \sim_E e = (d, c, \lambda, i)$, and use the appropriate restrictions of these relations as cell and edge equivalences on the input set of c.

Given a cell c in a network $\mathcal{N}$, we inductively define the *input tree* of c *of depth* n, denoted $I^n(c)$, so that each $I^n(c)$ has root c. The input tree of c of depth 1, $I^1(c)$, is $I(c)$, the input set of c. The input tree of c of depth $n-1$, denoted $I^{n-1}(c)$, has certain leaves at depth $n-1$. Each leaf d is associated with a cell $d_{\mathcal{N}}$ in $\mathcal{N}$. Join the input set of $d_{\mathcal{N}}$ to $I^{n-1}(c)$ at d. Having joined all of these input sets, the result is the new input tree $I^n(c)$, of depth n. We define $I^0(c)$ to be the tree consisting only of the cell c, with no edges. See Fig. 8 for an example.

The *infinite input tree* $I^\infty(c)$ is the direct limit of the input trees. This may or may not be an infinite network (even for finite networks $\mathcal{N}$), although if $\mathcal{N}$ is finite with no directed cycle, $I^\infty(c)$ is finite for all cells c of $\mathcal{N}$. Also, if $\mathcal{N}$ is locally finite, $I^\infty(c)$ is also locally finite for all cells c of $\mathcal{N}$.

Lemma 4.1. *For cells c from a locally finite network, each input tree of c of finite depth is finite. Further, the infinite input tree of such a cell c is countable, that is, it has a countable number of cells and edges.* (*Recall that our use of* "*countable*" *includes* "*finite*".)

Proof. This proof is routine, and omitted. ■

Given a relation $\sim$ on a network $\mathcal{N}$, we define the relation of *nth input tree equivalence under*

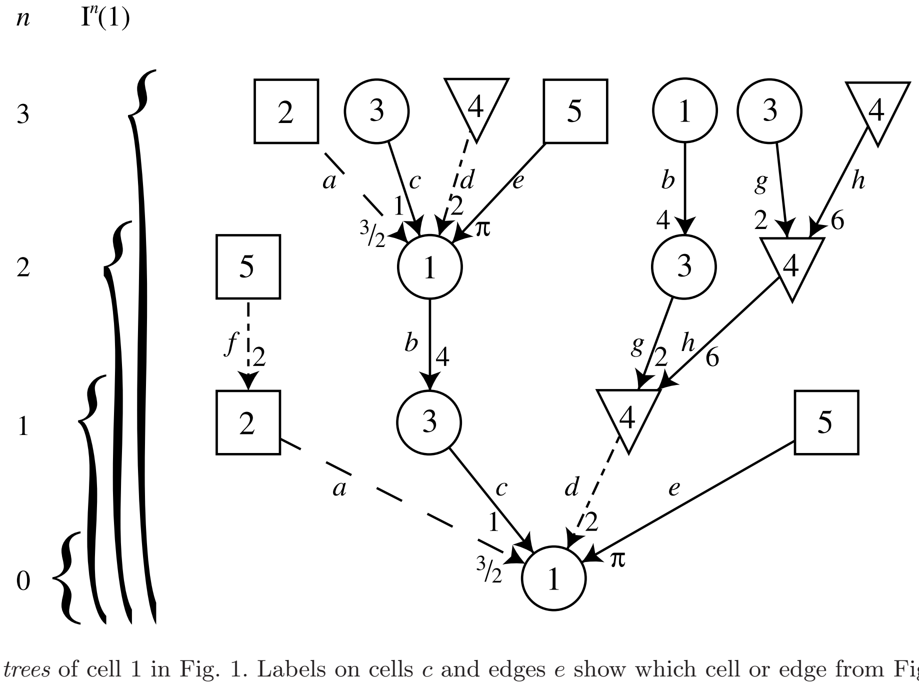


Fig. 8. *Input trees* of cell 1 in Fig. 1. Labels on cells c and edges e show which cell or edge from Fig. 1 gave c or e.

$\sim$ between cells of $\mathcal{N}$, denoted $c \cong^n_\sim d$ (for $n \in \mathbb{N} \cup \{\infty\}$), by $c \cong^n_\sim d$ when the input trees of c and d are equivalent: $I^n(c)/\!\!/\!\sim\ \cong_\sim I^n(d)/\!\!/\!\sim$. Since adding extra generations to two trees can only make them more different, it is clear that if $m \geq n$ then $\cong \sim^m \leq \cong^n_\sim$.

4.2. *Limits*

Given a sequence of input tree homomorphisms between the input trees of cells c and d, $(F^{(n)}_{c,d} : I^n(c) \to I^n(d))$ we now define its limit, the homomorphism $F^{(\infty)}_{c,d}$ between the infinite input trees of cells c and d.

Count, that is, uniquely enumerate from 1, the cells in $I^\infty(c)$ in increasing order of generation, with cells at the same generation enumerated in arbitrary order. (This is possible because the infinite input tree is countable, from Lemma 4.1.) Denote the cell enumerated $j \in \mathbb{N}$ as c_j. For each cell c_j, we define (by induction on j) a strictly increasing sequence of natural numbers, $n^{(j)}_i$ such that the subsequence $(d_{i,j}) = (F^{(n^{(j)}_i)}_{c,d})$ of $(F^{(n)}_{c,d})$ is constant on cells c_k for $k \leq j$. We then take $F^{(\infty)}_{c,d}(c_j) = d_{i,j} = F^{(n^{(j)}_i)}_{c,d}(c_j)$ (which is constant for all i).

To start the induction, first take $n^{(0)}_i = i$. Since the cells are enumerated from 1, this trivially satisfies the condition above.

Now assume we have such a sequence $(n^{(k)}_i)$ for all $k < j$. In particular, consider $n^{(j-1)}_i$: $(d_{i,j-1}) = (F^{(n^{(j-1)}_i)}_{c,d}(c_j))$ forms a sequence of cells in $I^\infty(d)$, all of which are at the same generation as the generation of c_j in $I^\infty(c)$, by Lemma 3.1. Since, by Lemma 4.1, there are only a finite number of such cells, this sequence must have a constant subsequence, $(d_{i,j}) = (F^{(n^{(j)}_i)}_{c,d}(c_j))$, with constant value d_j. (This need not be uniquely defined: if we have a "natural order" on the cells, we may take the first possible cell as the value of d_j, if not, we must understand that $F^{(\infty)}_{c,d}$ need not be uniquely defined.)

Thus by induction we have a sequence $n^{(j)}_i$ for all $j \in N$ such that $k, l \geq j \Rightarrow F^{(n^{(k)}_i)}_{c,d}(c_j) = F^{(n^{(l)}_i)}_{c,d}(c_j)\ \forall\, i \in \mathbb{N}$, giving well-defined values:

$$F^{(\infty)}_{c,d}(c_j) = \lim_{k\to\infty} \lim_{i\to\infty} F^{(n^{(k)}_i)}_{c,d}(c_j) = d_j$$

4.3. *Balanced equivalence relations*

An equivalence relation $\sim$ on a network $\mathcal{N}$ is said to be *balanced* if:

(1) $\sim$ is a refinement of $\sim_C$ and $\sim_E$, the cell and edge equivalence relations of $\mathcal{N}$. That is, $\sim\, \leq\, \sim_\mathcal{N}$.
(2) $c \sim d$ for cells c and d only where the input sets of c and d are equivalent under $\sim$. That is, $\sim\, \leq\, \cong^1_\sim$.
(3) $e \sim f$ for edges e and f only where their tails are equivalent, that is $e \sim f \Rightarrow \tau(e) \sim \tau(f)$. In general, we will define $\sim$ on cells, and let $e \sim f$ precisely when $e \sim_E f$ and $\tau(e) \sim \tau(f)$. Clearly, this produces the maximal $\sim$ on edges of a network for a given $\sim$ on its cells.

Figure 9 shows an example of a balanced equivalence relation.

We have defined a balanced equivalence relation as a relation on (the cells and edges of) a network. Previous work (for example, [Stewart *et al.*, 2003; Golubitsky *et al.*, 2004; Golubitsky *et al.*, 2005; Stewart, 2007]) has defined balanced equivalence relations as relations only on cells of a network. The notion of edge equivalence in the definition above is useful to us, but the definition we use is naturally related to the cell definition. The balanced cell equivalence relations are exactly the restrictions of the balanced network equivalence relations to the cells of the network; given any balanced cell equivalence relation $\sim$, we can extend it naturally to a balanced network equivalence relation $\approx$ by $e \approx f$ for edges e, f precisely when $e \sim_E f$ and the cells $\tau(e) \sim \tau(f)$. As remarked above, this natural extension $\approx$ is the maximal balanced network equivalence relation which is an extension of the cell relation $\sim$.

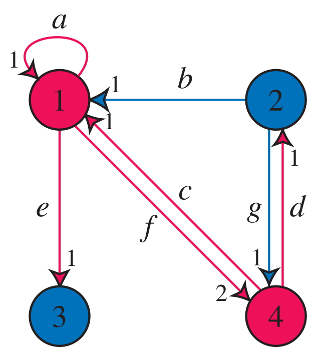


Fig. 9. A *balanced equivalence relation* on a graph. (Equivalence classes are represented by colors.)

Theorem 4.2. *Cells of a network $\mathcal{N}$ which are equivalent under a balanced equivalence relation $\sim$ on $\mathcal{N}$ have infinite input trees which are equivalent with respect to $\sim$. That is, $\sim \leq \cong^{\infty}_{\sim}$.*

Proof. Let $\sim$ be a balanced equivalence relation on a network $\mathcal{N}$, and c, d be two cells of $\mathcal{N}$ such that $c \sim d$. Then by definition of balanced equivalence relations, there is a isomorphism of bunched input sets $\phi^{(1)}_{c,d} : I^1(c)/\!\!/\!\sim \to I^1(d)/\!\!/\!\sim$ which preserves equivalence class under $\sim$. We extend this isomorphism inductively in the same way as the input tree is defined. Let n be given such that $\phi^{(n-1)}_{c,d}$ is defined. Start with $\phi = \phi^{(n-1)}_{c,d}$. For each leaf $\mathbf{c}'$ of $I^{n-1}(c)/\!\!/\!\sim$, make an arbitrary choice of contained cell $c' \in \mathbf{c}'$ (a cell of $I^{n-1}(c)$), which is in turn associated with some cell $c'_{\mathcal{N}}$ in $\mathcal{N}$. $\mathbf{c}'$ is taken by $\phi^{(n-1)}_{c,d}$ to some $\mathbf{d}'$: similarly, choose $d' \in \mathbf{d}'$ with associated cell $d'_{\mathcal{N}}$. Now adjoin the function $\phi^{(1)}_{c'_{\mathcal{N}},d'_{\mathcal{N}}}$ to ϕ at the leaf $\mathbf{c}'$. Once we have done this for all leaves of $I^{n-1}(c)$, we let $\phi^{(n)}_{c,d} = \phi : I^n(c)/\!\!/\!\sim \to I^n(d)/\!\!/\!\sim$. This gives a sequence of isomorphisms $\phi^{(n)}_{c,d} : I^n(c) \to I^n(d)$ with respect to $\sim$, where $\phi^{(n+1)}_{c,d}$ is an extension of $\phi^{(n)}_{c,d}$ for all n. The limit of this sequence is an isomorphism $\phi^{(\infty)}_{c,d}$ of infinite input trees bunched by $\sim$, which is well defined and preserves $\sim$, that is, a $\sim$-equivalence. Thus cells which are equivalent under a balanced equivalence relation have input trees which are equivalent with respect to that relation. ■

Corollary 4.3. *If $\sim$ is a balanced equivalence relation on a network $\mathcal{N}$, then $\cong^{\infty}_{\sim}$ is the same relation as $\sim$.*

Proof. Theorem 4.2 ensures that $c \sim d \Rightarrow c \cong^{\infty}_{\sim} d$, so we need only show $c \cong^{\infty}_{\sim} d \Rightarrow c \sim d$, which we do in the contrapositive. Suppose $c \not\sim d$. Then the roots of the trees $I^{\infty}(c)$ and $I^{\infty}(d)$, which are themselves c and d respectively, are in different equivalence classes under $\sim$. Hence there is no $\sim$-preserving isomorphism $I^{\infty}(c) \to I^{\infty}(d)$ (which would take $c \mapsto d$). Hence $c \not\cong^{\infty}_{\sim} d$, as required. ■

A type of limited converse to Theorem 4.2 is also true.

Lemma 4.4. *Given a network $\mathcal{N}$ and any equivalence relation $\sim$ on $\mathcal{N}$ which is a refinement of cell and edge equivalences, the relation $\cong^{\infty}_{\sim}$ is a balanced equivalence relation on $\mathcal{N}$.*

Proof. Take two cells c, d of $\mathcal{N}$ such that $c \cong^{\infty}_{\sim} d$. Then since $\cong^{\infty}_{\sim}$ is a refinement of $\cong^{1}_{\sim}$ (by the remark after the definition of $\cong^{n}_{\sim}$), $c \cong^{1}_{\sim} d$, as required. ■

Proposition 4.5. *If $\sim$ and $\approx$ are two relations on a network $\mathcal{N}$ such that $\sim \leq \approx$, then the relations $\cong^{\infty}_{\sim}$ and $\cong^{\infty}_{\approx}$ on $\mathcal{N}$ satisfy $\cong^{\infty}_{\sim} \leq \cong^{\infty}_{\approx}$.*

Proof. Let $\sim$ and $\approx$ be two relations on a network $\mathcal{N}$ such that $\sim \leq \approx$. Then $\sim$ is a refinement of $\approx$, that is $c \sim d \Rightarrow c \approx d$. Now suppose $c \cong^{\infty}_{\sim} d$, that is, $I^{\infty}(c)/\!\!/\!\sim \;\cong_{\sim} I^{\infty}(d)/\!\!/\!\sim$. We must show $c \cong^{\infty}_{\approx} d$, that is, $I^{\infty}(c)/\!\!/\!\approx \;\cong_{\approx} I^{\infty}(d)/\!\!/\!\approx$. This is given precisely by Proposition 3.8 for general trees $\mathcal{T}$ and $\mathcal{U}$. Let $\mathcal{T} = I^{\infty}(c)$ and $\mathcal{U} = I^{\infty}(d)$ for the required result. ■

As remarked earlier, the ordering $\leq$ on equivalence relations gives a lattice with minimal element $=$ and maximal element $\top$. With the same partial ordering, the set of balanced equivalence relations also forms a lattice with a minimal element (the relation $=$) and a maximal element (which we denote $\bowtie$). One proof of this was given in [Stewart, 2007]: this proof uses a number of sophisticated techniques to obtain the result for a class of not-necessarily-finite networks which include all locally finite networks. We give a more elementary proof which works for all networks.

Theorem 4.6. *For a given network $\mathcal{N}$, the set of balanced equivalence relations on $\mathcal{N}$ forms a complete lattice under the partial order given by refinement. In particular, $\mathcal{N}$ has a maximal and a minimal balanced equivalence relation.*

Proof. For any network $\mathcal{N}$, $=$ is (trivially) a balanced equivalence relation, so the set of balanced equivalence relations on $\mathcal{N}$ is nonempty and bounded below.

Take a nonempty set Y of balanced equivalence relations on $\mathcal{N}$. Then let $\approx$ denote the equivalence relation $\bigvee Y$, their join (taken in the lattice of equivalence relations on $\mathcal{N}$). We show that this relation $\approx$ is balanced: if $c \approx d$, then by definition there is some finite chain $c = c_0 \sim_1 c_1 \sim_2 \cdots \sim_n c_n = d$ with each $\sim_i \in Y$. In particular, $\sim_i$ is balanced. This means that the input sets of c_{i-1} and c_i are equivalent under $\sim_i$, so they are equivalent under $\approx$. Since equivalence under $\approx$ is an equivalence relation, the bunched input set of $c = c_0$ is isomorphic to that of $c_n = d$ with respect to $\approx$. Finally, we show that $\approx = \bigvee Y$ is a refinement of $\sim_{\mathcal{N}}$. Since all the relations in Y are refinements of the cell and edge equivalences $\sim_{\mathcal{N}}$, this $\sim_{\mathcal{N}}$ is an upper bound for Y. Hence $\bigvee Y \leq \sim_{\mathcal{N}}$, as required.

We have shown that the partially-ordered set of balanced equivalence relations on $\mathcal{N}$ has a minimal element, and that any nonempty subset Y of this set has a join. By Theorem 2.1, the set of balanced equivalence relations on $\mathcal{N}$ is a complete lattice, and so it has a maximal element. ∎

We have shown that there are two relations on $\mathcal{N}$ which are maximal and minimal balanced equivalence relations. As remarked above, the minimal equivalence relation $=$ is a balanced equivalence relation, and so it is the minimal balanced equivalence relation. We consider the structure of the maximal balanced equivalence relation, which we denote $\bowtie$. Since it is a balanced equivalence relation, equivalent cells have isomorphic bunched input trees with respect to the cell and edge type equivalence relations, by Theorem 4.2. In fact, this condition precisely categorizes $\bowtie$.

Theorem 4.7. *Given a network $\mathcal{N}$, the maximal balanced equivalence relation $\bowtie$ on $\mathcal{N}$ is given by: $c \bowtie d$ if and only if $I^\infty(c)/\!\!/\!\sim_{\mathcal{N}} \cong_{\sim_{\mathcal{N}}} I^\infty(d)/\!\!/\!\sim_{\mathcal{N}}$. That is, $\bowtie = \cong^\infty_{\sim_{\mathcal{N}}}$.*

Proof. The relation $\bowtie$ is balanced, so by Corollary 4.3, $\cong^\infty_{\bowtie} = \bowtie$. Meanwhile, as a balanced equivalence relation, $\bowtie$ refines $\sim_{\mathcal{N}}$, that is, $\bowtie \leq \sim_{\mathcal{N}}$. Hence $\cong^\infty_{\bowtie} \leq \cong^\infty_{\sim_{\mathcal{N}}}$, by Proposition 4.5. Also, by Lemma 4.4, $\cong^\infty_{\sim_{\mathcal{N}}}$ is balanced, and hence $\cong^\infty_{\sim_{\mathcal{N}}} \leq \bowtie$. So:

$$\bowtie = \cong^\infty_{\bowtie} \leq \cong^\infty_{\sim_{\mathcal{N}}} \leq \bowtie$$

Hence all of these relations are equal, and in particular, $\bowtie = \cong^\infty_{\sim_{\mathcal{N}}}$, as required. ∎

5. Algorithmic Method

This section contains the main result of this paper. We describe an algorithm which, given a finite network $\mathcal{N}$, finds the maximal balanced equivalence relation on $\mathcal{N}$ in polynomial time on total number of cells and edges in $\mathcal{N}$. We do this first in the case where $\mathcal{N}$ is unitary, that is, all multiplicities are equal to 1, in which case consideration of bunching is unnecessary, and then we extend it to the general case.

5.1. *Setup*

At each iteration t of the algorithm, each cell c and each edge e has a natural number attached, $n_t(c)$ and $n_t(e)$, (or just $n(c)$ and $n(e)$ when the "current" iteration is unambiguous). These shall be called the "colors" of c and e, as cells (and edges) with equal final values of $n(c)$ (and $n(e)$) are to be equivalent under the balanced equivalence relation we aim to define.

Let $|I_{\max}|$ represent the maximal size of an input set in $\mathcal{N}$.

Initial conditions: Start with $t = 0$. Let $n_0(c) = [[c]]$ and $n_0(e) = [[e]]$ for all cells c and edges e.

Constraint: $|[c]| \leq \max\{n_t(c)|c \in C\} \leq |C|$ and $|[e]| \leq \max\{n_t(e)|e \in E\} \leq |E|$ for all t. This arises because these numbers are always an enumeration of partitions of the cells and edges (giving the right hand inequalities). Each such partition is a subpartition of the previous one, so $\max\{n_t(c)|c \in C\}$ is increasing with t, and the first partition is derived from cell types, so $\max\{n_0(c)|c \in C\} = |[c]|$ (and similarly for edges).

Note: When enumerating quantities associated with cells or edges, follow the same order of cells or edges every time. That is, take an arbitrary "natural order" of cells and edges $c_1, c_2, c_3, \ldots$ such that the number $n_t(c_i)$ associated with each c_i follows the system: $n_t(c_1) = 1$, and $1 \leq n_t(c_{i+1}) \leq \max_{j \leq i}\{n_t(c_j)\} + 1$. This ensures that these numbers will be the same in subsequent enumerations of the same partitions.

5.2. *The algorithm*

(1) Phase 1:
   (a) Associate with each edge e the pair $(n_t(e), n_t(\tau(e)))$. ($|E|$ operations.)
   (b) Compare these pairs with each other and enumerate them. ($2|E|^2$ operations, at most.)
   (c) Let $n_{t+1}(e)$ be the number given in this way for the pair associated with e. ($|E|$ operations.)
   (d) Compare $n_t(e)$ and $n_{t+1}(e)$: if any of these numbers are different (that is, $n(e)$ has changed), or $t = 0$, continue (that is, go on to Phase 2). If not, stop. (At most $|E|$ operations.)

   Thus Phase 1 comprises at most $2|E|^2 + 3|E|$ operations.

(2) Phase 2:
   (a) Associate with each cell c the multiset $N_t(c) = \langle\!\langle n_t(e) | \eta(e) = c \rangle\!\rangle$. Recall that this gives one element of $N_t(c)$ for each e with head cell c, even when $n_t(e) = n_t(f)$ for two such edges e, f. (Note: writing the multiset in natural order of edges will ensure that it is written in an increasing fashion.) (Since each edge has some unique c at its head, this is precisely $|E|$ operations.)
   (b) Compare these multisets with each other and enumerate them. (Each of the $|C|$ multisets has at most $|I_{\max}|$ elements, so this is at most $|I_{\max}||C|^2$ operations, assuming the multisets are both increasing, which they can be by the note above.)
   (c) Let $n_{t+1}(c)$ be the number given in this way for the multiset associated with c. ($|C|$ operations.)
   (d) Compare $n_t(c)$ and $n_{t+1}(c)$: if any of these numbers are different, continue to the next iteration (that is, go back to Phase 1 with t increased). If not, stop. (At most $|C|$ operations.)

   Thus Phase 2 comprises at most $|I_{\max}||C|^2 + 2|C| + |E|$ operations.

When the algorithm terminates at iteration T, use the equivalence relation $c \sim d \Leftrightarrow n_T(c) = n_T(d)$ as the required balanced equivalence relation.

Notes

(1) Only one of the steps 1d and 2d is required. In fact, we can replace this step with a single comparison of the maximum $n(c)$ or $n(e)$ as appropriate, because at each iteration, the equivalence relation given by $n(c)$ (say) is a refinement of that at the previous iteration, so if any number changes, the maximum must also change. Thus Phase 1 can be bounded by $2|E|^2 + 2|E|(+1)$ operations, and Phase 2 by $|I_{\max}||C|^2 + |C|(+1)$, where the "+1" is used in the phase which incorporates its step d.

(2) In order to facilitate writing the multiset in step 2a in natural order of edges, let the input be structured as a list of blocks each consisting of a cell identifier followed by the edges with heads at that cell:

   Cell c
   Edge e with head cell c
   Edge f with head cell c
   ...
   Cell d
   Edge g with head cell d
   etc.

   Since every edge has exactly one head, each edge is listed exactly once. Let the input orders of the cells and edges be their natural orders.

Running time

Theorem 5.1. *This algorithm is quartic in $|E| + |C|$.*

Proof. At every iteration, by summing the operation counts shown in the algorithm, we can see that there are at most $|I_{\max}||C|^2+2|E|^2+3|E|+|C|+1 \le |E||C|^2+2|E|^2+3|E|+|C|+1 \le N^3+2N^2+3N+1$ operations, where $N = |E| + |C|$ is the length of the problem. Hence each iteration is cubic in the length of the problem, but better estimates of the time may be obtained from the above formula with knowledge of how the data set is broken down into edges and cells (for example, if the number of cells is held constant, each iteration becomes quadratic in the number of edges, and vice versa).

If part 2d is chosen (rather than part 1d), the algorithm can run for at most $|C|$ iterations, since at every iteration, except the last one the maximum $n(c)$ is strictly increasing, by Note 1. In fact, since the maximum $n(c)$ starts at $|[c]|$ and cannot get higher than $|C|$, the maximum number of iterations is $(|C| - |[c]|) + 1$ (the one additional stage is the one on which the algorithm makes no change to the $n(c)$s, and terminates.) By an analogous argument, if part 1d is chosen, the maximum number of iterations is $(|E| - |[e]|) + 2$. (The first iteration does not necessarily increase the maximum $n(e)$, since part 1d does not take effect until then.)

This gives the algorithm a total running time bounded above by:

$$(|E| + 2 - |[e]|)(N^3 + 2N^2 + 3N + 1)$$

or:

$$(|C| + 1 - |[c]|)(N^3 + 2N^2 + 3N + 1)$$

operations, depending on whether part 1d or 2d is included, respectively. These upper bounds are quartic in the size $|E| + |C|$ of the problem. ■

5.3. *Correctness*

It remains to show that the algorithm gives the correct result.

Theorem 5.2. *The relation given by the algorithm is the maximal balanced equivalence relation.*

Proof. We show that cells colored the same are infinite input tree equivalent, and then the converse. Recall that we are currently assuming all edges to have a multiplicity of 1, and hence $c \cong_{\sim}^{n} d$ precisely when $I^n(c) \cong_{\sim} I^n(d)$.

Firstly, we show that all cells colored the same by the algorithm are input set (first input tree) equivalent: as mentioned in Note 1, each n_t gives a relation which is a refinement of that given by n_t for smaller (earlier) t. Thus n_T gives a refinement of, in particular, n_1. If c, d are two cells such that $n_1(c) = n_1(d)$, then each edge with head at c must have a corresponding edge of the same edge type with head at d and the same cell type at its tail. Thus define a function $F_{c,d}^{(1)} : I^1(c) \to I^1(d)$ using these correspondences — $F_{c,d}^{(1)}$ is a first input tree (input set) equivalence.

Now we use induction to show that all cells of the same color are infinite input tree equivalent. Suppose all the cells colored the same by the algorithm are kth input tree equivalent, giving a balanced tree function $F_{c,d}^{(i)}$ between any identically-colored cells for $0 \leq i \leq k$. Then consider the $(k+1)$th input trees of two cells c, d of the same color. Extend $F_{c,d}^{(k)}$ to $F_{c,d}^{(k+1)}$: take pairs of identically colored cells c', d' at the kth level of the input trees of c, d respectively, where $F_{c,d}^{(k)}(c') = d'$, and adjoin $F_{c',d'}^{(1)}$ to $F_{c,d}^{(k+1)}$ at c'. By induction, the nth input trees of c and d are isomorphic for all n, and so the infinite input trees are also isomorphic, by Sec. 4.2.

Now we show the converse, that is, cells which are infinite input tree equivalent are colored the same by the algorithm.

It is clear that any two cells c, d which are input tree equivalent satisfy $n_0(c) = n_0(d)$, for in particular, they must have the same cell type, $[c] = [d]$, and so $[[c]] = [[d]]$. For any two cells c, d which are input tree equivalent, there is an input tree equivalence map between $I^{(\infty)}(c)$ and $I^{(\infty)}(d)$ which we shall denote $F_{c,d}^{(\infty)}$, as usual. Edges e, f which are identified by this equivalence must have the same edge type, and so $n_0(e) = n_0(f)$. For induction, suppose t is such that any two cells c, d that are input tree equivalent satisfy $n_t(c) = n_t(d)$, and further, for the same t, given an edge e in the input set of c and the corresponding edge $f = F_{c,d}^{(\infty)}(e)$ in the input set of d, $n_t(e) = n_t(f)$. Then take some pair of cells c, d that are input tree equivalent. The multisets $\langle\!\langle n_t(e) | \eta(e) = c \rangle\!\rangle$ and $\langle\!\langle n_t(e) | \eta(e) = d \rangle\!\rangle$ associated with the cells c and d at stage $t+1$ of the algorithm must be equal, since the edges must pair up by $F_{c,d}^{(\infty)}$. So $n_{t+1}(c) = n_{t+1}(d)$. Further, the pairs $(n_t(e), n_t(\tau(e)))$ and $(n_t(f), n_t(\tau(f)))$, associated with e and f respectively, are also equal; thus, $n_{t+1}(e) = n_{t+1}(f)$. Then by induction, $n_T(c) = n_T(d)$ where T is the final value of t used in the algorithm, and c and d are given the same color, as required. ■

Example

We illustrate this algorithm using the (homogeneous) unitary network described earlier in Fig. 2. It is worth noticing that this network has no nontrivial automorphisms, so a classical symmetry approach, treating cells c and d as equivalent precisely when there is an automorphism of the network which maps c to d, would give six equivalence classes of cells as "colors" — the trivial (balanced) equivalence relation of equality. Meanwhile, there are two equivalence classes of cells under input set equivalence. We shall see that the maximal balanced equivalence relation falls between these.

We take the natural order of the cells to be of numerical order, and that of the edges to be alphabetic.

At each iteration t, we show the numbers $n_t(c)$ and $n_t(e)$ as colors, to allow us to label the cells using numbers as usual. Table 1 shows the results.

5.4. *Multiplicity and bunching*

We now describe a more general algorithm, to take into account the possibility of nonunitary networks, that is, networks where edges may have any multiplicities. Obviously, if we wished only to allow edges to be paired with edges of the same multiplicity (ignoring bunching), we could replace Phase 1, step 1 with:

(1) (a) Associate with each edge e the triple $(|e|, n_t(e), n_t(\tau(e)))$, where $|e|$ is the multiplicity of edge e. ($|E|$ operations, as before.)

However, as shown in [Golubitsky *et al.*, 2005] and described earlier, when matching up input sets of two cells c and d, we wish to find an isomorphism between their bunched equivalents.

Table 1. An example of the algorithm to determine the maximal balanced equivalence relation for a unitary network.

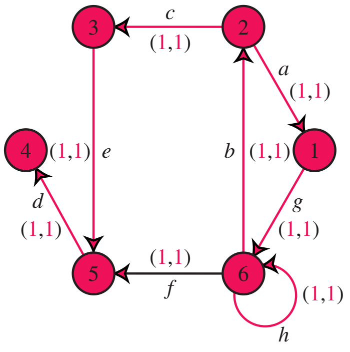
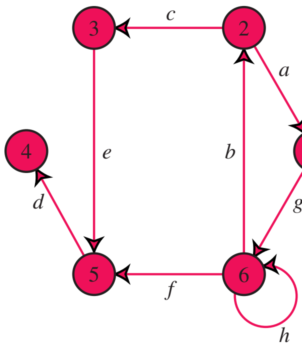
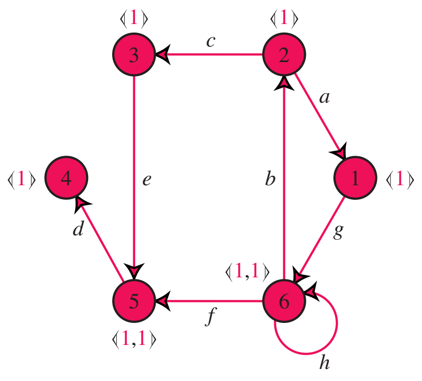
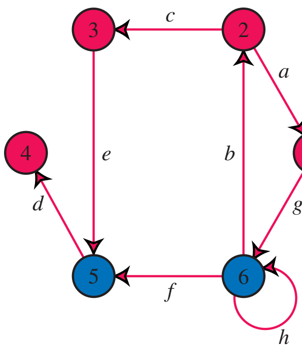
t	Phase	Steps	
0	1	(a)  1 2 3 4 5 6 a b c d e f g h (1,1)	(b) $(1,1) \mapsto 1$
		(c)  1 2 3 4 5 6 a b c d e f g h	(d) Continue (forced since $t = 0$)
	2	(a)  1 2 3 4 5 6 a b c d e f g h ⟨1⟩ ⟨1,1⟩	(b) $\langle 1 \rangle \mapsto 1$ $\langle 1, 1 \rangle \mapsto 2$
		(c)  1 2 3 4 5 6 a b c d e f g h	(d) Continue

Table 1. (*Continued*)

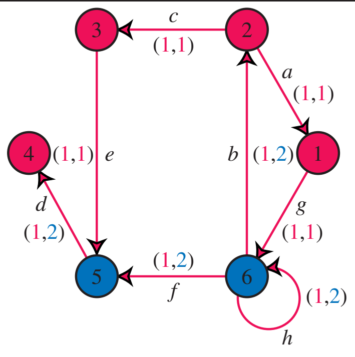
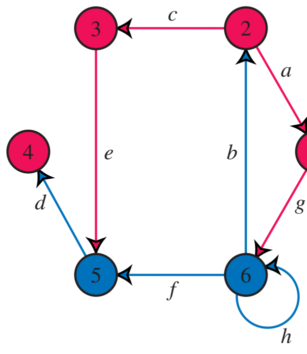
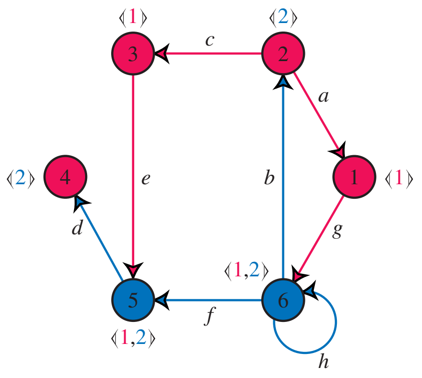
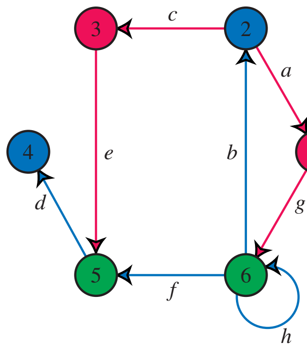
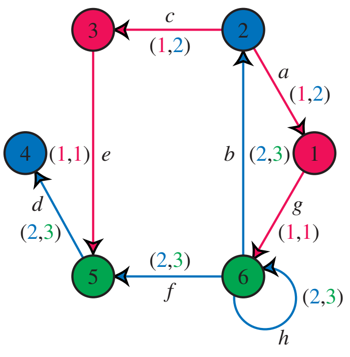
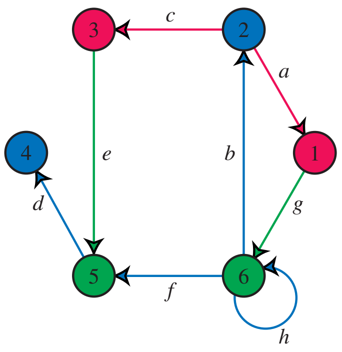
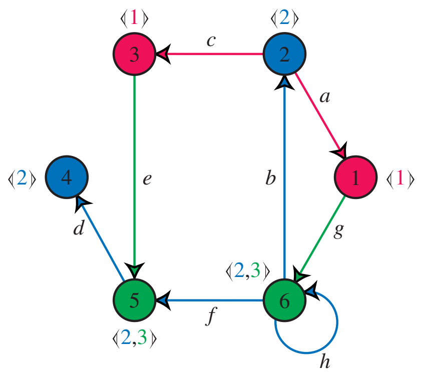
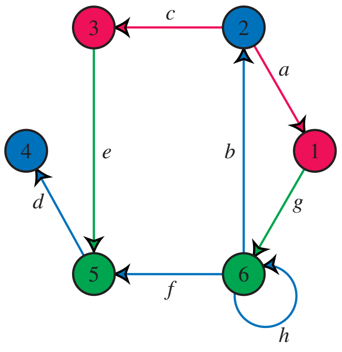
t	Phase	Steps	
1	1	(a)  1 2 3 4 5 6 a (1,1) b (1,2) c (1,1) d (1,2) e (1,1) f (1,2) g (1,1) h (1,2)	(b) $(1,1) \mapsto 1$ $(1,2) \mapsto 2$
		(c)  1 2 3 4 5 6 a b c d e f g h	(d) Continue
	2	(a)  ⟨1⟩ 3 ⟨2⟩ 2 ⟨2⟩ 4 1 ⟨1⟩ ⟨1,2⟩ 6 5 ⟨1,2⟩ a b c d e f g h	(b) $\langle 1 \rangle \mapsto 1$ $\langle 2 \rangle \mapsto 2$ $\langle 1,2 \rangle \mapsto 3$
		(c)  1 2 3 4 5 6 a b c d e f g h	(d) Continue

Table 1. (*Continued*)

t	Phase	Steps	
2	1	(a)  3 2 1 4 5 6 c a b d e f g h (1,2) (1,2) (1,1) (2,3) (2,3) (1,1) (2,3) (2,3) (2,3)	(b) $(1,2) \mapsto 1$ $(2,3) \mapsto 2$ $(1,1) \mapsto 3$
		(c) 	(d) Continue
	2	(a) 	(b) $\langle 1 \rangle \mapsto 1$ $\langle 2 \rangle \mapsto 2$ $\langle 2,3 \rangle \mapsto 3$
		(c) 	(d) Cell numbers have not changed since $t = 1$. Stop.

In order to do this, instead of the replacement to Phase 1, step a above, replace Phase 2, step a with:

(2) (a) For each cell c consider the set of edge numbers $T_c = \{n_t(e) | \eta(e) = c\}$. Now for each n in T_c, consider the sum $\lambda_n = \sum_{n_t(e)=n, \eta(e)=c} |e|$. Now associate with c the set $\{(n, \lambda_n) | n \in T_c\}$. Each of these pairs precisely corresponds to the bunch $\mathbf{e}$ where $n(e) = n$ for $e \in \mathbf{e}$. This set may be constructed in a more direct way in practice: keep the representation of the set in

Table 2. An example of the algorithm to determine the maximal balanced equivalence relation for a general network.

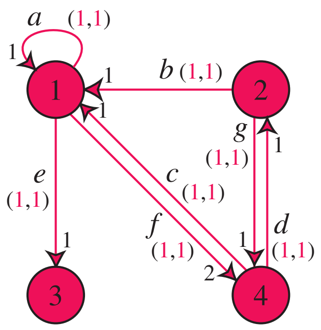
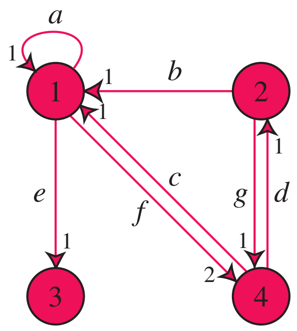
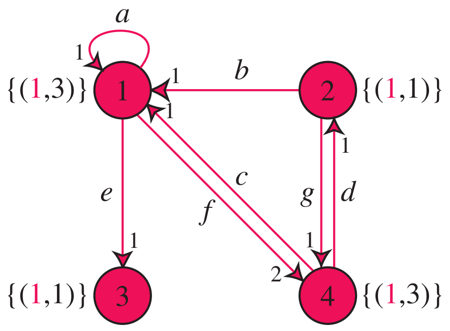
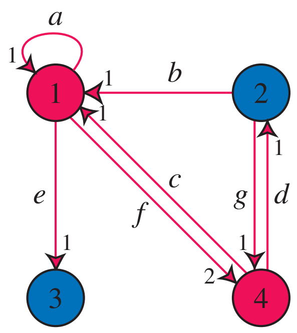
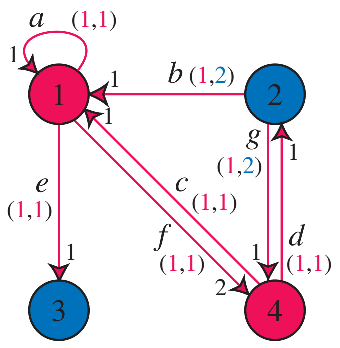
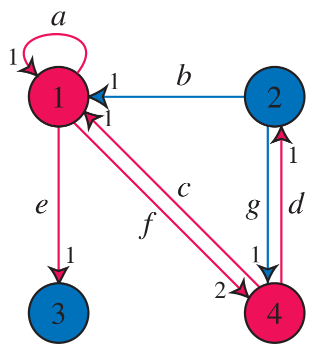
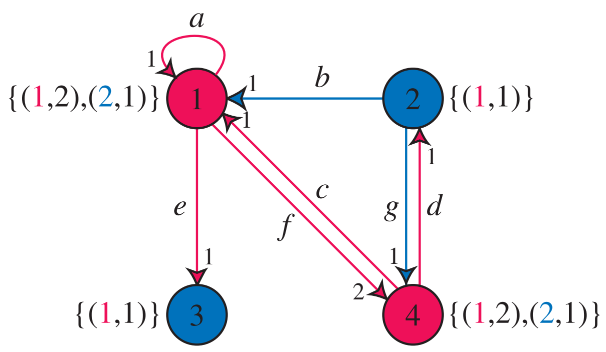
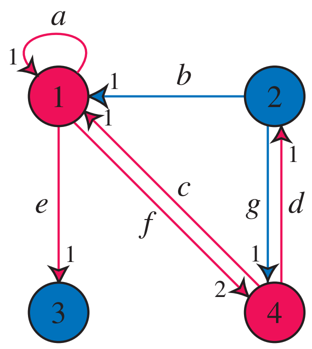
t	Phase	Steps		
0	1	(a)		(b) $(1,1) \mapsto 1$
		(c)		(d) Continue (forced since $t = 0$)
	2	(a)		(b) $\{(1,1)\} \mapsto 1$ $\{(1,3)\} \mapsto 2$
		(c)		(d) Continue

Table 2. (*Continued*)

t	Phase	Steps
1	1	(a)  a (1,1) b (1,2) c (1,1) d (1,1) e (1,1) f (1,1) g (1,2) 1 2 3 4 (b) $(1,1) \mapsto 1$ $(1,2) \mapsto 2$ (c)  (d) Continue
	2	(a)  a b c d e f g {(1,2),(2,1)} {(1,1)} {(1,1)} {(1,2),(2,1)} 1 2 3 4 (b) $\{(1,2),(2,1)\} \mapsto 1$ $\{(1,1)\} \mapsto 2$ (c)  (d) Cell numbers have not changed since $t = 0$. Stop.

order of increasing n. For each edge e with the given head cell c, find the pair in the set associated with $c = \eta(e)$ whose first component is $n_t(e)$. Add $|e|$ to the second component. If we reach a pair in the set whose first component is greater than $n_t(e)$ without finding one whose first component is exactly $n_t(e)$, insert $(n_t(e), |e|)$ before this greater pair. (Still at most $|E|$ operations.)

This replacement makes no difference to running time estimates.

This replacement treats a set of equivalent edges with equivalent tails and the same head as a single edge with multiplicity equal to the sum of the multiplicities of the original edges for the purpose of matching input sets, while leaving the original network unchanged — so if two cells c, d have $n_t(c) = n_t(d)$ but $n_{t+1}(c) \neq n_{t+1}(d)$ then edges from c and d will not be treated as part of the same "bunch" at iteration $t + 1$, even though they were treated as such at iteration t. Thus this change to the algorithm provides exactly the desired result.

Example

We give a short example to demonstrate this algorithm in use, calculating the maximal balanced equivalence relation of the network shown in Fig. 9. As before, we take numerical and alphabetical natural orders of cells and edges, respectively. Table 2 shows the results.

Acknowledgments

Some of the details of the algorithm were suggested and refined during a productive conversation with Daan Krammer, for which the author is very grateful.

The research in this paper was undertaken while studying at the University of Warwick. The majority of this research was submitted for the degree of Master of Science [Aldis, 2005]; the exception being Theorem 4.6, which was proved while studying for the degree of Doctor of Philosophy. The paper was also edited and prepared for submission during the latter course of study; this course was partially supported by a grant from EPSRC. Both of these courses of study have benefited from the invaluable supervision of Professor Ian Stewart.

References

Aldis, J. W. [2005] "A polynomial time algorithm to determine maximal balanced equivalence relations," Masters thesis, University of Warwick.

Davey, B. A. & Priestley, H. A. [1990] *Introduction to Lattices and Order*, 1st edition (Cambridge University Press, Cambridge).

Golubitsky, M., Stewart, I. N. & Nicol, M. [2004] "Some curious phenomena in coupled cell networks," *J. Nonlin. Sci.* **14**, 207–236.

Golubitsky, M., Stewart, I. N. & Török, A. [2005] "Patterns of synchrony in coupled cell networks with multiple arrows," *SIAM J. Appl. Dyn. Syst.* **4**, 78–100.

Golubitsky, M. & Stewart, I. N. [2006] "Nonlinear dynamics of networks: The groupoid formalism," *Bull. Amer. Math. Soc.* **43**, 305–364.

Stewart, I. N., Golubitsky, M. & Pivato, M. [2003] "Symmetry groupoids and patterns of synchrony in coupled cell networks," *SIAM J. Appl. Dyn. Syst.* **2**, 609–646.

Stewart, I. N. [2007] "The lattice of balanced equivalence relations of a coupled cell network," *Math. Proc. Camb. Phil. Soc.* **143**, 165–183.