Online Supplement to
Collaboration and Multitasking in Networks: Architectures, Bottlenecks and Capacity

Itai Gurvich
Kellogg School of Management, i-gurvich@kellogg.northwestern.edu,

Jan A. Van Mieghem
Kellogg School of Management, vanmieghem@kellogg.northwestern.edu,

---

Proof of Theorem 1: We first argue that given I, we can construct a network with I activities and $K = 3 + \sum_{i=3}^{I-1} i$ resources that has a complete collaboration graph and such that each resource has only two activities, i.e, $\sum_j A_{kj} = 2$ for all $k \in \mathcal{K}$. We take the collaboration graph of the BC+ network as the starting point for this construction. This is a complete graph with three nodes: 1,2 and 3. With $\mathcal{R}(\{1\}) = \{1,2\}$, $\mathcal{R}(\{2\}) = \{2,3\}$ and $\mathcal{R}(\{3\}) = \{3,1\}$ and has $K = 3$ resources as needed. Now assume we construct the required network for all network sizes less than or equal to $I-1$. We next add an activity (numbered I) as follows. For each node i in the current graph (a total of $I-1$ nodes) create a (new) resource k_i, assign it to activity i and add it to $\mathcal{K}$ (before this step $K = 3 + \sum_{i=3}^{I-2} i$). Assign all these new resources to the new activity I. That is, $\mathcal{R}(\{I\})$ is the set of these newly added resources. Each of the added resources k_i has two activities (activities i and I) and activity I is connected with an edge to each of the nodes of step $I-1$. After this step, then, we have a network with a complete collaboration graph, where each resource has two activities and $K = 3 + \sum_{i=3}^{I-2} i + (I-1) = K = 3 + \sum_{i=3}^{I-1} i$. Finally, set the service time and arrival rate to each activity i to $m_i = 1$ and $\alpha_i = 1/2$. Set $P = 0$. Then, $\rho^{BN} = \max_{k\in\mathcal{K}} \sum_j A_{kj} = 1$. However, the only feasible configuration vectors are the identity vectors so that to satisfy $(\sum_l \pi(e^l)e^l)_j = \lambda_j m_j = 1/2$ we must $\pi(e^j) = 1/2$ and, in turn, that $\rho^{\text{net}} = \sum_j \pi(e^j) = I/2$. ■

Proof of Theorem 2: We prove this result for multiserver networks and the single-server case follows as a special case. We start with the sufficiency. Consider the polyhedron

$$\Xi = \Big\{ x \geq 0 : \sum_i A_{ki} x_i \leq n_k,\ k \in \mathcal{K} \Big\}.$$

By definition, if the SPP (17) has a solution $\rho^{\text{BN}} \leq 1$, we have that $\sum_i A_{ki}\lambda_i m_i \leq \rho^{\text{BN}} n_k$ or, re-writing,

$$\sum_i A_{ki} \frac{\lambda_i m_i}{\rho^{\text{BN}}} \leq n_k,\ k \in \mathcal{K}.$$

In particular, $x^{\mathrm{BN}} = (\lambda_1 m_1/\rho^{\mathrm{BN}}, \dots, \lambda_I m_I/\rho^{\mathrm{BN}})$ is in the convex polyhedron Ξ and can be expressed as a convex combination of its extreme points. From the assumption that the extreme points are integer it follows that the extreme points of Ξ are integer valued for any integer right hand side. In particular, the extreme points of Ξ are feasible configuration vectors; see (18)). Thus, x^{BN} can be written as a convex combination of configurations: there exists $\pi \geq 0$ where $e'\pi = 1$ and

$$\sum_j \pi^j a^j = x^{\mathrm{BN}},$$

where a^j are feasible configuration vectors. Setting $\hat{\pi} = \rho^{\mathrm{BN}}\pi$ we have, as required, that

$$\sum_j \hat{\pi}^j a^j = \lambda m, \text{ and } \rho^{\mathrm{net}} = \sum_l \hat{\pi}_l = \rho^{\mathrm{BN}}.$$

For the necessity, assume that the polyhedron Ξ has a non-integral extreme point x^*. Choose (α, P, μ) such that $\lambda m = x^*$. Then, x^* cannot be expressed as a (non-trivial) convex combination of integer vectors in Ξ. In particular, there exists no π such that $\pi \geq 0$, $e'\pi \leq 1$ and with $\sum_i \pi_i a^i = 1$ for configuration vectors a^i. In particular, it must be the case that $\rho^{net} > 1 \geq \rho^{BN}$.

Finally, note that if the adjacency matrix A is Totally Unimodular (TUM) it follows that the extreme points of Ξ are integer valued (for any integer right hand side); see (Schrijver 1998, Corollary 19.2a). ■

Proof of Corollary 1: Notice first that we may ignore resources that have a single activity. Those resources must be assigned in full to their single activity and it suffices to consider the residual network. Here, each resource has exactly two activities with an edge connecting the two activities in the graph. In particular, each edge is associated with a single resource. There can be two resources collaborating on the two activities but in that case they can be treated as a single resource. Thus, we may assume without loss of generality that there is a one to one mapping between edges and resources. With each resource corresponding to an edge and each activity to a node, the matrix A^T (the transpose of A) is then (by the assumption of the corollary) the incidence matrix of a bi-partite graph and is hence totally unimodular; see e.g. (Schrijver 1998, page 273). The transpose of a totally unimodular matrix is itself totally unimodular and we conclude that A is a TUM matrix. In turn, by Theorem 3 the network features no unavoidable idleness. ■

The following will be used in subsequent proofs.

An auxiliary synchronization graph for networks with nested architectures: Recall that for a network to have a nested collaboration architecture it is not necessary that there be no cycles in the collaboration graph. It is only required that all cycles are nested-sharing cycles. For nested architectures we can construct an auxiliary acyclic graph that has the useful property that activities at the same "level" of the graph do not share resources – we will refer to this acyclic graph as the synchronization graph. This graph is a tool rather than a conceptual entity.

Recall that a nested-sharing cycle is a set of l connected nodes (activities) $i_0,\ldots,i_l$ such that

$$\mathcal{S}(i_k,i_j)\subseteq\mathcal{S}(i_m,i_j) \text{ for all } k,m,j\in\{1,\ldots,l\}: j>m>k.$$

We refer to i_0 as the highest rank activity in the cycle, i_1 the second rank etc. Following standard terminology we say that a simple path in a graph between nodes i and j is a set of *distinct* nodes $i,i_1,\ldots,i_l,j$ such that each two consecutive nodes are connected by an edge. Given a network, we construct the synchronization graph as follows:

0. **Initialization:** Set $\ell=0$ and $\mathcal{C}^0=\emptyset$. Add a fictitious node r and set $d_r=0$. (think of the root as an activity with $\lambda_r=0$ that uses all K resources.)

1. If $\mathcal{C}^\ell=\mathcal{I}$ stop. Otherwise, consider the maximal sharing between an activity outside the graph and one in the graph, i.e,

$$\mathcal{O}=\max\{i\notin\mathcal{C}^{\ell-1}, j\in\mathcal{C}^{\ell-1}:|\mathcal{S}(i,j)|\}.$$

Set $\mathcal{O}=0$ if there are no such i and j.

2. $\mathcal{O}>0$: Pick an activity $i_\ell\notin\mathcal{C}^{\ell-1}$ with $|\mathcal{S}(i_\ell,j)|=\mathcal{O}$ for some $j\in\mathcal{C}^{\ell-1}$. If taking an activity from a nested-sharing cycle take the highest-rank activity in that cycle not yet in the graph (if there are multiple nested-sharing cycles with activities not yet in the graph, take an activity from a cycle with the largest number of nodes).

Pick a node $j_\ell\in\arg\max_{j\in\mathcal{C}^{\ell-1}:|\mathcal{S}(i_\ell,j)|=\mathcal{O}} d_j$, add the edge (i_ℓ,j_ℓ) and set $d_{i_\ell}=d_{j_\ell}+1$. (we are connecting i_ℓ to the activity with the greatest distance from the root among those that share resources with i_ℓ). Stop if there exist $j,k\in\mathcal{C}^{\ell-1}$ with $|\mathcal{S}(i_\ell,j)|=|\mathcal{S}(i_\ell,k)|=\mathcal{O}$ but $\mathcal{S}(i_\ell,j)\neq\mathcal{S}(i_\ell,k)$.

3. $\mathcal{O}=0$ (there are no activities i,j one in the graph and the other out of it that share resources): pick an arbitrary $i_\ell\notin\mathcal{C}^{\ell-1}$ and add to the graph connecting it via an edge to the root. If adding an activity from a nested-sharing cycle take the one with the highest rank (if there are multiple nested-sharing cycles with activities not yet in the graph, take an activity from a cycle with the largest number of nodes). Set $d_{i_\ell}=1$.

Example A.1 Consider the network in Figure 9 together with its collaboration graph. The collaboration graph contains only nested-sharing cycles. The outcome of the algorithm applied to this network is as in Figure 10 (we removed here the fictitious root). ∎

Since the choice of the edge to add in step 2 of the algorithm is arbitrary there can be multiple synchronization graphs but, importantly, the following holds.

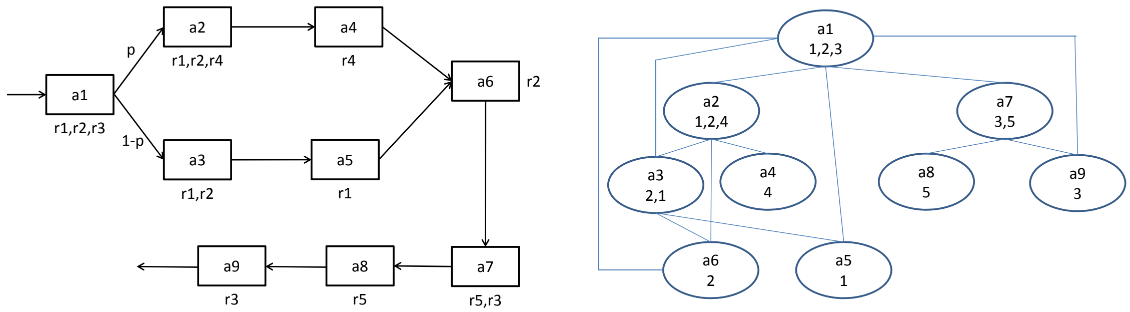

Figure 9 **A network with nested hierarchical architecture**

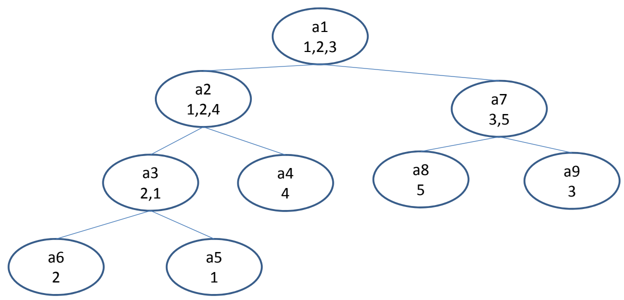

Figure 10 **A synchronization graph for the network in Figure 9**

Lemma A.1 *(1) If the network has a nested collaboration architecture the algorithm generates a graph with I nodes and no cycles and, (2) in this case, activities in the same level of the graph (i.e, with the same parameter d) do not share resources ($\mathcal{S}(i,j)=\emptyset$ if $d_i=d_j$).*

Proof: If the algorithm stops after I steps then all nodes were added with a single edge and no cycles were formed. Suppose, towards contradiction, that the algorithm stops after $\ell < I$ steps. In this step a node i_ℓ is added and there are j_ℓ and k_ℓ with $\mathcal{O}=|\mathcal{S}(i_\ell,j_\ell)|=|\mathcal{S}(i_\ell,k_\ell)|$ and $\mathcal{S}(i_\ell,j_\ell)\neq\mathcal{S}(i_\ell,k_\ell)$. We claim that the cycle containing i_ℓ, k_ℓ and j_ℓ must be a non-nested cycle. Suppose that these activities are, in fact, part of a nested-sharing cycle. Let us further assume that k_ℓ was added to the graph after j_ℓ (the other case is argued identically). By assumption, $\mathcal{O}=|\mathcal{S}(i_\ell,j_\ell)|=|\mathcal{S}(i_\ell,k_\ell)|$. Since this is a nested-sharing cycle and k_ℓ has a lower rank than j_ℓ we have that $\mathcal{S}(j_\ell,i_\ell)\subseteq\mathcal{S}(k_\ell,i_\ell)$ and, in turn, $\mathcal{S}(i_\ell,j_\ell)=\mathcal{S}(i_\ell,k_\ell)$ which is a contradiction to the stopping rule. We may thus conclude that, if the network has a nested architecture, the algorithm ends with a tree that includes I nodes.

We argue next that if the architecture is nested then i and j with $d_i=d_j$ must have $\mathcal{S}(i,j)=\emptyset$. The case that i and j are not part of a cycle is trivial as, if $d_i=d_j$ and $\mathcal{S}(i,j)\neq\emptyset$, we would have in fact found a cycle containing i and j in the collaboration graph. To argue the case that i and j are part of a nested-sharing cycle, let ℓ be the first step in which a node i_ℓ is added with the property that $d_{i_\ell} \leq d_j$ for some $j \in \mathcal{C}^{\ell-1}$ with $\mathcal{S}(i_\ell, j) \neq \emptyset$. Let $k \neq j$ be such that (i_ℓ, k) is the edge that is added to the graph with node i_ℓ (if $k = j$ we would have $d_{i_\ell} = d_j + 1 > d_j$). Then, $d_j \geq d_{i_\ell} = d_k + 1$. In particular $d_k < d_j$. Note that $\mathcal{S}(j, k) \neq \emptyset$ as they are both part of a nested-sharing cycle containing j. Since i_ℓ is the first node added with the required property, the fact that $d_k < d_j$ implies that k was added to the graph before j (and has higher rank in the nested-sharing cycle containing both). By the definition of nested-sharing cycles we must then have that $\mathcal{S}(k, i_\ell) \subseteq \mathcal{S}(j, i_\ell)$ and, in particular, that $|\mathcal{S}(k, i_\ell)| \leq |\mathcal{S}(j, i_\ell)|$. Recall that also $d_j > d_k$ so that, when adding i_ℓ we would have added the edge (i_ℓ, j) instead of (i_ℓ, k). ■

Proof of Theorems 4 and 6: First, note that Theorem 4 is a special case of Theorem 6 with the staffing vector set to be the vector of ones. We divide the proof into two parts. In the first we treat nested architectures and in the second we treat weakly non-nested architectures.

Nested architectures: A known sufficient condition for the total unimodularity of the matrix A is that it (or a permutation of its rows) has the consecutive ones property; see (Schrijver 1998, Example 7, Chapter 19). We next prove that we can re-label the resources and permute the rows so that the 1s in each column (corresponding to an activity) appear consecutively.

Our starting point is the synchronization tree constructed above. We first re-organize the tree. We make sure that at every level of the tree the nodes with the least number of sons are far from the root. Formally, i is a parent node of j (and j the son of i) if there is an edge between them and $d_j = d_i + 1$. Returning to the example we used before, the graph in Figure 11(LHS) would be re-organized into the one on the RHS.

Proceeding with this example, we can now re-label resources following depth-first-search to traverse the tree. We first visit activity $a8$. This activity has the single resource 5 – we re-label this resource as 1 (i.e, $5 \to 1$). We then proceed to activity $a9$ and re-label 3 as 2. At this point labels 1 and 2 are already taken and the next available label is 3. In activity $a7$, $3, 5$ is replaced with $2, 1$ (or 1,2 for convenience of display) following the re-labeling already done in the son nodes of $a7$. We then visit node $a6$ and replace $2 \to 3$ and in activity $a5$ $1 \to 4$. In activity $a3$ we re-label based on the son nodes $a5$ and $a6$. For activity $a4$, the next available resource number is 5 so we re-label $4 \to 5$ and follow accordingly in activities $a2$ and finally $a1$. By the end of this procedure we have re-labeled the resources $(5, 3, 2, 1, 4) \to (1, 2, 3, 4, 5)$. This re-labeling guarantees the consecutive 1's property: $a1$, for example, uses resources 2,3 and 4 (previously 1,2,3), $a2$ uses 3,4,5 (previously 1,2,4), etc.

The following is the formalization of the re-labeling algorithm:

Initialize $num = 0$ and $z_0 = 0$. Each resource has a tuple containing its original number k, its current label $\ell(k)$ (which is initialized to k), and a binary variable $v(k)$ which is 0 initially and set to 1 once k is labeled. We take the following actions in step l of the depth first search:

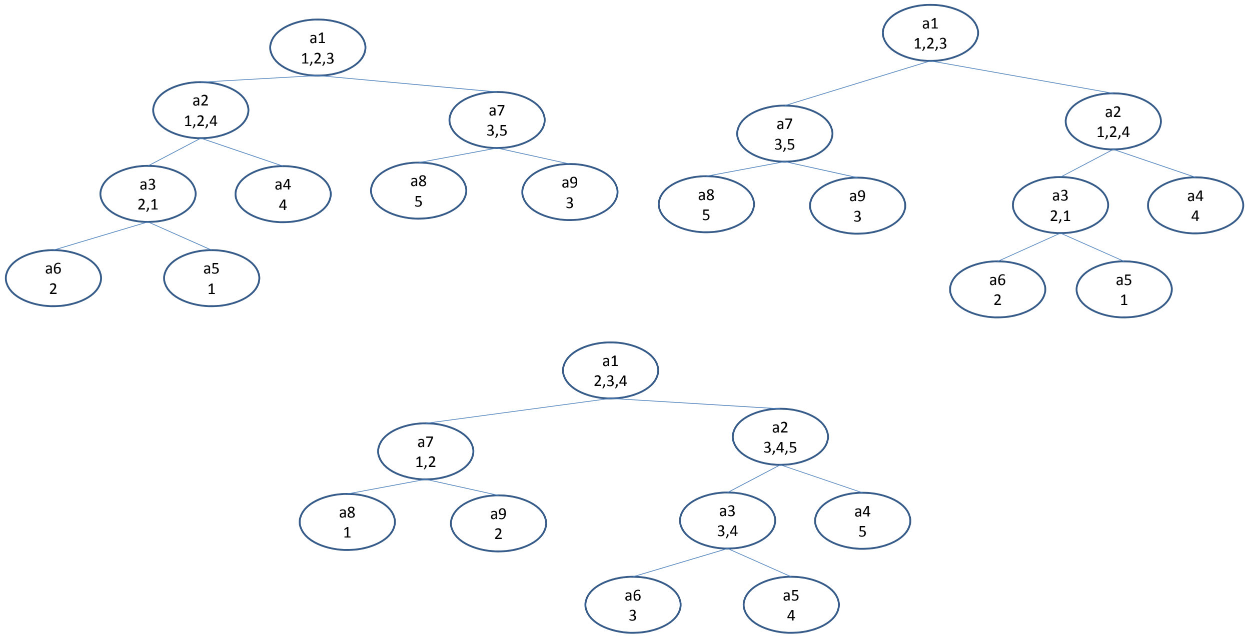

Figure 11 (LHS) A synchronization graph for the network in Figure 9 and (RHS) re-organized version and (BOTTOM) re-labeled resources

1. If the node is a leaf (corresponding to activity i say), we label all unlabeled resources in this node in an arbitrary order starting with the numbers $num+1,\ldots,num+|\mathcal{R}(\{i\})|$. We advance $num \leftarrow num+|\mathcal{R}(\{i\})|$. For each labeled resource k, we write $\ell(k)$ for its new label and set $v(k)=1$.

2. If node i is not a leaf:

2a. If node i has a resource k that has not yet been marked (i.e. $v(k)=0$): if i is on the left of the root assign it the number z_l-1 (and change $z_l \leftarrow z_l-1$). If i is on the right of the root, label $\ell(k)=num+1$, set $v(k)=1$ and advance $num \leftarrow num+1$.

2b. Order the resources in each activity in increasing order of their labels. If after completing step $2a$ there is a gap in the labels of resources in activity i (there are resources $k,l \in \mathcal{R}(\{i\})$ such that $\ell(k)>\ell(l)+1$ but no $\kappa \in \mathcal{R}\{i\})$ with $\ell(\kappa)=\ell(l)+1$)) we take the following actions: Let k (with label $\ell(k)$) be the first resource after the gap. Let j be a son of i such that $k \in \mathcal{S}(i,j)$ (by Lemma A.1 there can be at most one such son node). Re-label all resources in the sub-tree rooted in j by shifting them by $-\ell(k)$. Repeat as long as there are gaps.

To illustrate step 2b consider Figure 12. The top graph on the left is the original one and the bottom graph on the left is the one obtained after applying all steps except for step 2b on the root node $a1$. Note that in the root node there is now a gap (between 2 and 4). In this last step we take the sub-tree rooted at $a5$ and shift all labeling by -4, thus creating the two new labels -1 and 0. All labels in the graph are now consecutive.

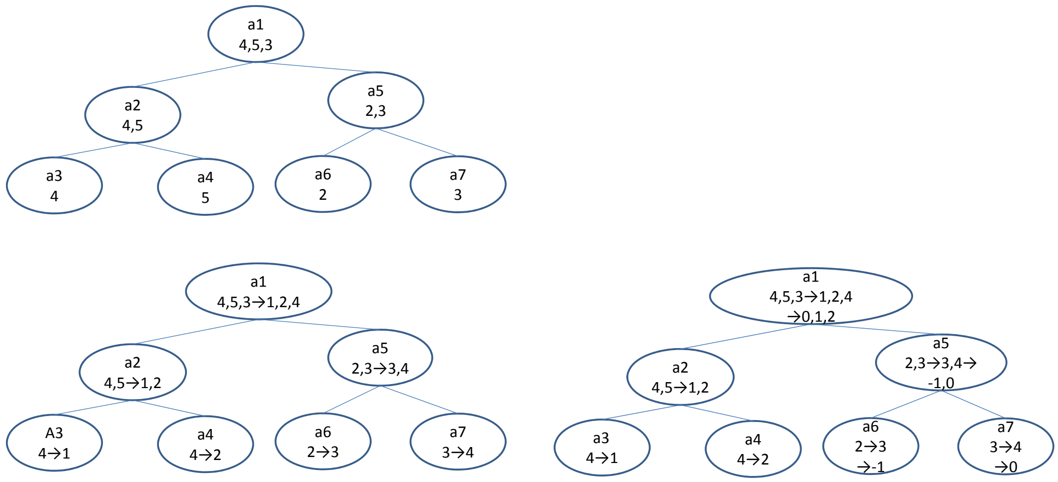

Figure 12 **Relabeling example**

To argue that the resulting labeling has consecutive labels we perform induction on the step number. This is obviously true for the first visited leaf. Since no two leafs in the synchronization tree have shared resources, when a leaf is visited no resource is already labeled. Assuming that for all activities visited in step $l \leq k-1$ resources are consecutively numbered, the algorithm preserves this property. Let i be the activity visited in the k^{th} step: if step 2b is not applied to node i, it means there is no gap and the consecutive labeling is inherited from the son nodes because new resources are added to the left (if the node is to the left of the root) or right (if the node is on the right of the root). If step 2b is applied in this node then the resource numbers are merely shifted and hence, by the induction assumption, all son nodes preserve the consecutive-labels property.

Notice that the fact (recall Lemma A.1) that the synchronization tree does not have nodes with shared resources in the same level, is used in step 2b.

Finally, permuting the rows of A according to the labeling we created, each column in the graph (corresponding to each activity) will have consecutive ones. Recall that this guarantees that the matrix A is totally unimodular which concludes the proof for nested architectures.

Weakly non-nested architectures: We will show that if an architecture is weakly non-nested, we can alter the network in a way that preserves the value of ρ^{BN} and can only increase ρ^{net} but has a nested collaboration architecture. This will imply, by the first part of this theorem, that there is no unavoidable idleness.

We start with an example. Consider a network of 3 activities and 4 resources as in the collaboration graph in Figure 13(LHS)–each circle corresponds to an activity and the required resources are listed below the activity's label.

This network contains a non-nested cycle. It is weakly non-nested because resource 4 is shared by all activities in the cycle. The bottleneck in this network is trivially resource 4 with $\rho_4 = \rho^{\text{BN}} =$

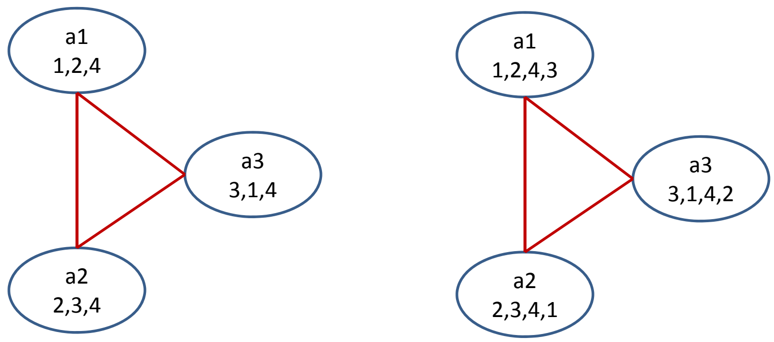

Figure 13 Transforming a weakly non-nested network into a nested one without changing ρ^{BN}.

$\sum_{j=1}^{3}\lambda_j m_j$ (and $\rho_2,\rho_1,\rho_3\le\rho_4$). If we add each of the resources 1,2 and 3 (each, notice, is assigned initially to 2 activities) to activities to which they are not assigned, we obtain the network with the collaboration graph on Figure 13(RHS). The resulting network is trivially nested. Importantly, this action does not affect the theoretical utilization which remains $\rho^{\mathrm{BN}}=\rho_i=\sum_{j=1}^{3}\lambda_j m_j,\ i=1,\dots,4$ and it can only increase ρ^{net} as, by assigning more resources to activities we can only shrink the family of feasible configuration vectors.

The new network we constructed is nested so that $\rho^{\mathrm{BN}}=\rho^{\mathrm{net}}$. In particular, we can construct an allocation π that achieves $\rho^{\mathrm{net}}=\rho^{\mathrm{BN}}$. In this special example, positive weights are given only to the identity vectors $e^i,\ i=1,2,3$.

To generalize this argument, fix a network with a weakly non-nested collaboration architecture. As in the above example, we first transform all weakly non-nested cycles into nested ones.

Fix a weakly non-nested cycle of activities $\mathcal{C}\subseteq\mathcal{I}$. Let k^* be the focal resource of this cycle: the resource that is shared by all activities in the cycle. If there are multiple cycles in which k^* is the focal resources, $\mathcal{C}$ is taken to be the one with the most activities. We can also assume there is a single such resource for $\mathcal{C}$. If there are two we can treat them, without loss of generality as the same resource. Let $\mathcal{K}(\mathcal{C})=\{k\in\mathcal{K}: A_{kj}=1,\ \text{for some } j\in\mathcal{C}\}$ be the set of resources that participate in at least one activity in the cycle.

By definition $j\in\mathcal{C}$ if and only if $A_{k^*j}=1$. We distinguish between two types of resources associated with this cycle:

(i) Resources that participate in two activities or more in the cycle. The set of these is given by $\mathcal{K}_{\ge 2}(\mathcal{C}):=\{k\in\mathcal{K}(\mathcal{C}):\sum_{j\in\mathcal{C}}A_{kj}\ge 2\}$.

We claim that a resource $k\in\mathcal{K}_{\ge 2}(\mathcal{C})$ cannot have activities $j\notin\mathcal{C}$. Indeed, suppose that there exist $k\in\mathcal{K}_{\ge 2}(\mathcal{C})$ and $j_0\in\mathcal{I}$ such that $A_{kj_0}=1$ but $A_{k^*j_0}=0$. Since k participates in two activities in the cycle $\mathcal{C}$, there must exist two activities $j_1,j_2\in\mathcal{C}$ such that j_0,j_1 and j_0,j_2 are in the graph.

Note that because all activities in $\mathcal{C}$ share a resource we can assume without loss of generality that the activities j_1, j_2 are consecutive activities in the cycle (otherwise we can re-label the activities). Thus, we have identified a non-nested cycle $\bar{\mathcal{C}}$ (with more activities than $\mathcal{C}$). Note that $\bar{\mathcal{C}}$ must be a non-nested cycle. If it were nested than the smaller cycle $\mathcal{C}$ would also be nested. Moreover, it is strongly non-nested because $A_{k^* j_0} = 0$. This would be a contradiction to the assumption that all non-nested cycles are weakly non-nested.

(ii) Resources that participate in one activity in the cycle $\mathcal{K}_1(\mathcal{C})$.

We do nothing for resources $k \in \mathcal{K}(\mathcal{C}) \setminus \mathcal{K}_{>2}(\mathcal{C})$. Since we have argued that for $k \in \mathcal{K}_{>2}(\mathcal{C})$, $\sum_{j \notin \mathcal{C}} A_{kj} = 0$, we can alter the network by assigning k to each of the activities $j \in \mathcal{C}$ with (initially) $A_{kj} = 0$ and still have $\sum_j A_{kj} \lambda_j m_j \leq \sum_j A_{k_{\mathcal{C}} j} \lambda_j m_j$ so that the value of $\rho^{\text{BN}} = \max_k \rho_k$ does not change. Note that the resulting cycle is nested. Any resource that appears twice appears now in all activities of $\mathcal{C}$ so that, in any order, the condition (15) holds. Repeating the same for each such weakly non-nested cycle, the network is transformed into a nested network. For this network $\rho^{\text{net}} = \rho^{\text{BN}}$. Since, by assigning resources to more activities we only shrink the family of configuration vectors this, in particular, implies for the original network that $\rho^{\text{BN}} = \rho^{\text{net}}$ which concludes the proof. ■

Proof of Lemma 2: Let $\mathcal{C} = i_0, \ldots i_l$ be the shortest amongst the strongly non-nested sharing cycles. A segment of the cycle is a subset of consecutive activities in the cycle. Since $\mathcal{C}$ is strongly non-nested it can be divided into non overlapping segments (the end point of one segment can serve as a starting point for the next) such that for each segment there is a resource k that is shared by all activities in this segment. There must be at least two such segments since the cycle is, by assumption, strongly non-nested.

Note that there cannot be another strongly non-nested cycle in the graph that has nodes in two distinct segments of the cycle $\mathcal{C}$. Otherwise $\mathcal{C}$ would not be the shortest strongly non-nested cycle. Also, there can not be a nested-sharing cycle (or a weakly non-nested cycle) with nodes in two distinct segment because by definition both nested and weakly non-nested cycles require the existence of a resource that is shared by all activities in the cycle.

We conclude that there are no edges in the collaboration graph with end points in distinct segment of this cycle. We can then assume, without loss of generality, that each segment has one edge (and two activities). Indeed, if there are three activities there will be an edge between each two of them because they share a resource and we can drop one activity. Thus, we have found a simple cycle. ■

Proof of Theorem 5: By Lemma 2 the network contains at least one simple non-nested cycle. Let M be the (odd) number of nodes in the cycle (it is also the number of edges).

Choose λ and m such that $\lambda_j m_j = 1/2$ for each activity on the cycle. Set $\lambda_j m_j = 0$ for all other activities in the network. Recall that a cycle $i_1, \ldots, i_l$ is simple non-nested if each two activities connected by an edge share a resource that is not used in any other activity in the cycle. With the above parameters we can assume that there is one such resource per edge (if there are multiple we can treat them as the same resources) and a total of M resources assigned to activities in the cycle.

Each resource that defines an edge on the cycle has two activities with a total load of 1 and is thus a bottleneck. Since at most $\lfloor (M-1)/2 \rfloor < M/2$ of the M activities on this cycle can be processed in parallel and each activity uses 2 of these resources there is no feasible configuration set $\mathcal{A}$ with $\mathbf{BN} \subseteq \mathcal{R}(\mathcal{A})$. The condition of Lemma 3 is trivially satisfied and we can conclude that the network features unavoidable idleness. ■

Proof of Lemma 3: Suppose that $\rho^{\text{net}} = \rho^{\text{BN}} = 1$. Let (π, ρ^{net}) be a solution to the SPPC (i.e, $\sum_{\mathcal{A} \in \mathbb{C}} a(\mathcal{A})\pi(\mathcal{A}) = \lambda m$. and $\sum_{\mathcal{A} \in \mathbb{C}} \pi(\mathcal{A}) = \rho^{\text{net}}$).

Since $\sum_i A_{ki} a_i(\mathcal{A}) \in \{1, 0\}$ for any feasible configuration set $\mathcal{A}$, we have that

$$\begin{aligned} \rho_k &= \sum_i A_{ki}(\lambda_i m_i) = \sum_i A_{ki} \Big(\sum_{\mathcal{A} \in \mathbb{C}} a(\mathcal{A})\pi(\mathcal{A})\Big)_i \\ &= \sum_{\mathcal{A} \in \mathbb{C}} \pi(\mathcal{A}) \sum_i A_{ki} a_i(\mathcal{A}) = \sum_{\mathcal{A} \in \mathbb{C}, k \in \mathcal{R}(\mathcal{A})} \pi(\mathcal{A}). \end{aligned} \tag{26}$$

Moreover, if $\mathcal{A}$ is such that $\pi(\mathcal{A}) > 0$ then it must be the case that $\mathbf{BN} \subseteq \mathcal{R}(\mathcal{A})$. Indeed, for all $k \in \mathbf{BN}$, the right hand side of (26) is $\rho^{\text{net}} = \rho^{\text{BN}} = 1$. Thus, if there exist $k, l \in \mathbf{BN}$ and $\mathcal{A}$ with $\pi(\mathcal{A}) > 0$ such that $k \in \mathcal{R}(\mathcal{A})$ but $l \notin \mathcal{R}(\mathcal{A})$ then we would have $\sum_{\mathcal{A} \in \mathbb{C}: l \in \mathcal{R}(\mathcal{A})} \pi(\mathcal{A}) < 1 = \rho^{\text{BN}}$.

In turn, if $\rho^{\text{net}} = \rho^{\text{BN}} = 1$ there exists a family $\mathbb{C}(\mathbf{BN}) \subseteq \mathbb{C}$ such that $\mathbf{BN} \subseteq \mathcal{R}(\mathcal{A})$ for each $\mathcal{A} \in \mathbb{C}(\mathbf{BN})$ and $\sum_{\mathcal{A} \in \mathbb{C}(\mathbf{BN})} \pi(\mathcal{A}) = 1$.

Finally, for each i, $(\sum_{\mathcal{A} \in \mathbb{C}} a(\mathcal{A})\pi(\mathcal{A}))_i = \lambda_i m_i$ (recalling that $a(\mathcal{A})$ is a binary vector) so that $\pi(\mathcal{A}) \leq \min_{i \in \mathcal{A}} \lambda_i m_i$. We conclude that, if $\rho^{\text{BN}} = \rho^{\text{net}}$ there must exist a family of subsets $\mathbb{C}(\mathbf{BN})$ of $\mathbb{C}$ such that

$$1 = \sum_{\mathcal{A} \in \mathbb{C}(\mathbf{BN})} \pi(\mathcal{A}) \leq \sum_{\mathcal{A} \in \mathbb{C}(\mathbf{BN})} \min_{i \in \mathcal{A}} \lambda_i m_i.$$

In particular, if

$$\sum_{\mathcal{A} \in \mathbb{C}: \mathbf{BN} \subseteq \mathcal{R}(\mathcal{A})} \min_{i \in \mathcal{A}} \lambda_i m_i < 1,$$

it must be the case that $\rho^{\text{net}} > \rho^{\text{BN}} = 1$.

For the second part of the lemma, arguing as before we obtain that

$$\rho_k = \sum_{\mathcal{A}:k\in\mathcal{R}(\mathcal{A})} \pi(\mathcal{A}) \le \rho^{\text{net}} \left(\sum_{\mathcal{A}:k\in\mathcal{R}(\mathcal{A})} \min_{i\in\mathcal{A}} \lambda_i m_i \right).$$

(Recall that $\rho^{\text{net}} \geq 1$ in the assumptions of the lemma.) Then, for each $k \in \mathbf{BN}$

$$\rho^{\text{BN}} = \rho_k \le \rho^{\text{net}} \max_{l\in\mathbf{BN}} \left(\sum_{\mathcal{A}:l\in\mathcal{R}(\mathcal{A})} \min_{i\in\mathcal{A}} \lambda_i m_i \right),$$

which completes the argument. ■

Proof of Lemma 4: The proof is by construction. Let ρ^{BN} be the solution of the SPP. Let $c^{0,\theta}$ be the configuration vector that has as its i^{th} entry

$$c_i^{0,\theta} = \frac{1}{\rho^{\text{BN}}} \left\lfloor \lambda_i^\theta m_i \right\rfloor.$$

(note that since we assume throughout the paper that $\lambda_i m_i > 0$ for at least one i, we have that $\rho^{\text{BN}} > 0$.) Let $\pi(c_i^{0,\theta}) = \rho^{\text{BN}}$. Also, let $\mathfrak{n}^{i,\theta}$ be the vector that has $\min_{k:k\in\mathcal{R}(\{i\})} n_k^\theta$ in its i^{th} entry and 0 otherwise. Set

$$\pi(\mathfrak{n}^{i,\theta}) = \frac{1}{\min_{k:k\in\mathcal{R}(\{i\})} n_k^\theta} \left(\lambda_i^\theta m_i - \left\lfloor \lambda_i^\theta m_i \right\rfloor \right).$$

Note that all vectors c^0 and $(\mathfrak{n}^i,\ i \in \mathcal{I})$ are feasible configuration vectors since they satisfy $\sum_i A_{ki} c_i^{0,\theta} \le n_k^\theta$ for all k and $\sum_l A_{kl} \mathfrak{n}_l^{j,\theta} = \min_{l:l\in\mathcal{R}(\{j\})} n_l \le n_k^\theta$ for all k. Finally, note that

$$\pi(c^{0,\theta})c^{0,\theta} + \pi(\mathfrak{n}^{i,\theta})\mathfrak{n}^{i,\theta} = \lambda_i^\theta m_i$$

and

$$\pi(c^{0,\theta}) + \sum_i \pi(\mathfrak{n}^{i,\theta}) = \rho^{\text{BN}} + \sum_i \frac{1}{\min_{k:k\in\mathcal{R}(\{i\})} n_k^\theta} \left(\lambda_i^\theta m_i - \left\lfloor \lambda_i^\theta m_i \right\rfloor \right) =: \rho^\theta.$$

Thus, the θ-s SPPC has a feasible solution $(\pi^\theta, \rho^\theta)$ with

$$|\rho^{\text{BN}} - \rho^\theta| \le \sum_i \frac{1}{\min_{k:k\in\mathcal{R}(\{i\})} n_k^\theta}$$

Finally, since λ is strictly positive by assumption and each resource is assigned to at least one activity we must have that $n_k > 0$ to have a feasible solution for the SPP. Since $n_k^\theta = \theta n_k \to \infty$ as $\theta \to \infty$, we conclude that $|\rho^{\text{BN}} - \rho^\theta| \to 0,$ as $\theta \to \infty$. ■

Proof of Theorem 7: The proof is straightforward given the definitions and Theorem 4 for the no-flexibility case. Specifically, given $(x^{\mathrm{BN}}, \rho^{\mathrm{BN}})$ as in the statement of theorem (i.e, that solve (23)), consider the following problem

$$\begin{array}{rl} \text{minimize} & \rho \\ \text{s.t.} & \sum_i A_{k,(iG)} x_{iG}^{BN} \leq \rho, \text{for all } k \in \mathcal{K}, \end{array}$$

This can be interpreted as the SPP corresponding to an artificial network with activities $\{(iG)\}$ and with arrival rate x_{iG}^{BN} to activity (iG). Trivially, this problem has ρ^{BN} as its optimal solution. The collaboration architecture of this artificial network is nested by assumption. By Theorem 4, there exists $\pi \geq 0$ such that $\sum_{\mathcal{A}\in\mathbb{C}} \pi(\mathcal{A}) = \rho^{\mathrm{BN}}$ and $\sum_{\mathcal{A}\in\mathbb{C}} a(\mathcal{A})\pi(\mathcal{A}) = x^{\mathrm{BN}}$. Thus, $\sum_G \sum_{\mathcal{A}\in\mathbb{C}} (a(\mathcal{A})\pi(\mathcal{A}))_{iG} = \sum_G x_{iG}^{\mathrm{BN}} = \lambda_i m_i$ where the last equality follows directly from the SPP. ∎

The following provides a weaker sufficient condition than the one in Theorem 7. We let $\rho(x) = \max_k \sum_k A_{k,(iG)} x_{iG}$.

Lemma A.2 *Fix λ and let (x^{BN}, ρ^{BN}) be an optimal solution to the SPP with $\rho^{BN}(\lambda) \leq 1$. Suppose that x^{BN} can be written as a sum of non-negative vectors $x^1, \ldots x^\ell$ each of which induces a nested extended collaboration architecture and such that $\sum_{l=1}^{\ell} \rho(x^l) \leq 1$. Then, $\rho^{net}(\lambda) \leq 1$.*

Proof: For each m we can construct as in the proof of Theorem 7 a probability vector such that $\sum_a \pi(a) = \rho^m$ where ρ^m is the value of the static planning problem for x^m. A probability vector $\breve{\pi}$ is then constructed by setting $\breve{\pi}(a) = \sum_m \pi^m(a)$. by assumption $\sum_a \breve{\pi}(a) \leq 1$ and $\sum_a a\breve{\pi}(a) = x^{\mathrm{BN}}$. ∎

Proof of Lemma 6: Identically to Lemma 3 it is proved that if the SPP has an optimal solution $\rho^{\mathrm{BN}} = 1$ and

$$\sum_{\mathcal{A}\in\mathbb{C}:\mathbf{BN}\subseteq\mathcal{R}(\mathcal{A})} \min_{(i,G)\in\mathcal{A}} \lambda_i m_i < 1,$$

then the network features unavoidable bottleneck idleness. Let $\mathbb{C}^F$ be the family of feasible configuration sets after the addition of the extended activity (i,G) as in the statement of the lemma. Under the conditions of the lemma the extended activity (i,G) cannot participate in any covering of $\mathbf{BN}^F = \mathbf{BN}\bigcup\{k\}$. In particular, $\{\mathcal{A}\in\mathbb{C}^F : \mathbf{BN}^F \subseteq \mathcal{R}(\mathcal{A})\} \subseteq \{\mathcal{A}\in\mathbb{C} : \mathbf{BN} \subseteq \mathcal{R}(\mathcal{A})\}$ so that under the condition of the lemma

$$\sum_{\mathcal{A}\in\mathbb{C}^F:\mathbf{BN}^F\subseteq\mathcal{R}(\mathcal{A})} \min_{(i,G)\in\mathcal{A}} \lambda_i^F m_i \leq \sum_{\mathcal{A}\in\mathbb{C}:\mathbf{BN}\subseteq\mathcal{R}(\mathcal{A})} \min_{(i,G)\in\mathcal{A}} \lambda_i^F m_i < 1$$

and by the first part of the lemma the network features unavoidable idleness. ∎