Spaceflight Multi-Processors with Fault Tolerance and Connectivity Tuned from Sparse to Dense [1]

Laurence E. LaForge [2]
Larry.LaForge@The-Right-Stuff.com

Jeffrey R. Moreland [2]
Jeff.Moreland@The-Right-Stuff.com

M. Sami Fadali [2]
Fadali@IEEE.org

The Right Stuff of Tahoe, Incorporated
The Right Place, 3341 Adler Court
Reno, NV 89503-1263 USA

+1.775.322.5186

Electrical Engineering Department
University of Nevada
Reno, NV 89557-0153 USA

+1.775.784.6951

Abstract—We describe a novel generation of multi-processor architectures, with fault tolerance and connectivity tuned from sparse to dense. *Multivariate feasible regions* quantify how such architectures minimize channel cost and latency, and maximize throughput and fault tolerance. Key to designs which optimally exploit these feasible regions: a software-automated catalog of results from the mathematics of connectivity. For example, discoveries about *Hamming graphs* set the stage for algorithms that configure the corresponding topologies. We introduce a new theorem that explicates the separability-covering duality of Hamming graphs, together with a new, efficient algorithm for recognizing and labeling Hamming graphs of arbitrary radix and dimension. Previously reported algorithms run slower, and emphasize Hamming graphs with radix two. Grid computing applications that benefit from tunable fault tolerance and connectivity include i) *design* of multi-processors, coupled via vertical cavity surface emitting lasers (VCSELs); ii) *auto-configuration* of self-healing mobile *ad hoc* networks (MANETs) via digital radio-frequency channels.

This is an in-depth paper targeting engineers, computer scientists, or applied mathematicians with a background in quantitative, dependable computing.[3] We provide a tutorial illustrating how the mathematics of connectivity solves seven practical problems, present seven new theoretical results, and pose nine open challenges. Two complementary works appear in these proceedings:

– Addressing a general audience with an overview of our work, the diagrams and informal narrative of "Multi-Processors by the Numbers" [44] unfold how feasible regions govern multi-processor design and operation, and elaborate grid computing as mentioned in the *Abstract* above.
– Technologists and engineers may also be interested in "Vertical Cavity Surface Emitting Lasers for Spaceflight Multi-Processors" [43], our broadly-scoped report on VCSELs as enablers for tunable architectures.

Table of Contents

1. *Tunable* Multi-Processor Topologies

Advances in *commodity* circuit and communication technologies can – and should – enable breakthroughs for fault tolerant, autonomous, evolvable spacecraft avionics. Especially over the past decade:

a) *Technologists* from Alkalai [1] to Zorian [17] have repeatedly underscored the case for *bottom-up* changes in how we approach the design and operation of electronics systems in general, spacecraft avionics in particular.

b) *System architects* have echoed a *top-down* vision, typically invoking mission benefits. Their voices, in fact, frequently reflect technology-driven (albeit high-level) views of what is both achievable and desirable [4], [5], [27], [34].

1. 0-7803-9546-8/06/$20.00 © 2006 IEEE. In *Proceedings, 2006 IEEE Aerospace Conference*. Big Sky, Montana, March 4-11, 2006.
2. Sponsored by the United States Missile Defense Agency (contract HQ0006-05-C-0028).
3. Session 7.08: Fault Tolerance, Autonomy, and Evolvability in Spacecraft Avionics. Final version 3, paper #1353 (23 pages).

Table 1: Basic terminology, cross-referenced and as used in this paper. Also see any text, such as [7] or [10], on graph theory.

adjacency, neighbor(s); adjacency list; pp. 4, 17	The *adjacency* of a *vertex* x is the set $\partial(x)$ of *vertices* (the *neighbors* of x) with which x shares an edge. Common practice represents a *graph* as a set of *adjacency lists*. Also see *topology*
antipodal chord	Use: Line 8, Algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$, p. 5. Definition: p. 17. Synonym: *diametric* p. 17
ball: open, closed; Table 1	In any *metric space* S, an *open* (*resp. closed*) *ball* of radius r and center x equals points of S whose *distance* from x is less than (*resp.* less than or equal to) r
bipartite graph; p. 9 *two-colorable*	*Graph* G is *bipartite* if its *vertices* comprise two classes, and no *edge* belongs to just one class. Alternatively, G is *two-colorable*. Equivalently, G contains no odd *cycles*
cardinality; Table 2	Short for "number of elements". Saves 7 characters per use. *Cf. primality* (of a *factorization*)
clique; Table 2, pp. 8, 13	*Strict graph* with each of $\frac{1}{2}(n^2-n)$ possible *edges*. Synonyms: *hyperedge*, *complete graph*
cover; packing; matching	Definitions: p. 7. Also note qualifiers: *minimum*, *maximum*, *maximal*, (*in*)*complete*
connected: (sub)graph or *component; connectivity;* Eqn. (2), p. 5	G is *connected* if every pair of *vertices* in G comprise the *endpoints* of at least one *path* in G; alternatively, G contains a *spanning tree*. The *vertex (edge) connectivity* is the fewest number of *vertices (edges)* whose deletion results in a lone *vertex*, or (p. 5) two or more *components*
cycle; $G_{\text{H-H}}(n, 1)$*;* Table 2	An *edge* joining the *endpoints* of a *path* whose *pathlength* is at least two
degree; p. 5	Number of *edges* to which a *vertex* belongs. In *strict graphs*: the number of *vertex neighbors*
diameter, DIAM; p. 5 *radius,* RAD; p. 5	$\text{DIAM} = \max_x \text{ECC}(x)$ and $\text{RAD} = \min_x \text{ECC}(x)$ are the maximum *resp*. minimum *eccentricities*. By Thm 2.4 of [10]: $\text{RAD}(G) \le \text{DIAM}(G) \le 2 \cdot \text{RAD}(G)$
eccentricity, ECC; Table 2	The maximum *edge distance* from one *vertex* x to some other. $\text{ECC}(x) = \max_y \lvert x, y \rvert$
edge, *graph distance*; p. 7	Least *pathlength* $\lvert x, y \rvert = \min_{P(x \ldots y)} \lvert P(x \ldots y) \rvert$ between *vertices* x and y. We prefer *edge distance* rather than *graph distance* to accent the difference with *Hamming distance* $\lvert x, y \rvert_H$
factor; factorization; p. 7	A *prime factorization* has minimum *primality* (p. 7); *i.e.*, the fewest number of *factors* (p. 7)
fault(s) TITLE *tolerance,* p. 4 *model,* p. 5 *worst-case,* p. 5 *fractional,* p. 9 *probabilistic* p. 9 *healthy* p. 4 *node,* p. 4 *channel* p. 4	A *fault* is a condition which renders the *node* or *channel* untrustworthy, hence (in this paper) unusable. A network instantaneously subjected to faulty *nodes* and *channels tolerates* them if the remaining *healthy nodes* and *channels* form a *quorum*. Also in this paper: we assume that faulty nodes have been diagnosed, and that hardware and software can deny their connection to *healthy* nodes. The *fault tolerance* is the maximum number of faults that an n-node network can tolerate. Under a *graph* model, the tolerance f to faulty *nodes* is at most the *tolerance* to faults in *channels*, or any combination of *faults* in *nodes* and *channels*, for $n > f+1 > 1$. Throughput this paper we therefore employ a conservative model of faults in nodes. The *worst-case* (nodal) network *fault tolerance* f equals one less than the *(vertex) connectivity* $f+1$ of the corresponding graph. The *fractional fault tolerance* f / n equals the fault tolerance divided by the number of nodes. The *fractional connectivity* $(f+1) / n$ is only slightly less than the *fractional fault tolerance*, and is more convenient for comparing the cost of worst-case fault tolerance with the *probabilistic* cost of tolerating a constant fraction of faults that are distributed, say, with Bernoulli probability p
(di)graph: directed, weighted	*Vertex, edge, arc, hyperedge:* pp. 4, 5, 13. *Planar*: pp. 10, 14. *Cf. adjacency*. *Strict graph*: each *edge*'s *endpoints* are distinct (no loops) and appear in but one *edge*. *Cf. multi-graph* p. 9
Hamiltonian; p. 8	*Path* or *cycle spanning* all *vertices*
Hamming distance, graph	Definitions: p. 13. Also see *Gray code*, p. 13
(hyper)cube; mesh	Definitions: pp. 13, 20. Qualifiers: *dimension*, (*majorized*) *radix*. *Cf. hyperedge*, *j-edge.*
(hyper)edge, separator	Definitions: pp. 13, 14. Also see: *hyperedge matching, hyperfactorization*, p. 14
interior-disjoint, IDJ; p. 5	Two *paths* are *interior-disjoint* (*IDJ*) if, apart from their *endpoints*, they do not intersect
metric space, *distance; distance space;* Table 1	Real-valued *distance* $\lvert \cdot, \cdot \rvert \ge 0$ mapping all pairs from a given set, such that: a) $\lvert x, y \rvert = 0$ if and only if $x = y$; b) symmetry: $\lvert x, y \rvert = \lvert x, y \rvert$; c) triangle inequality: $\lvert x, y \rvert \le \lvert x, y \rvert + \lvert x, y \rvert$
order, size; p. 10	*Vertex resp. edge cardinality*

Table 1: Basic terminology, cross-referenced and as used in this paper. Also see any text, such as [7] or [10], on graph theory.

path, endpoint, (path)length; pp. 4, 17	*Endpoint vertex* x, or union $P(x \ldots z)$ of *path* $P(x \ldots y)$ with *edge* (y, z), such that y and $z \notin P(x \ldots y)$ are *endpoints* of $P(x \ldots y)$ *resp.* $P(x \ldots z)$. The *(path)length* of P is its *size*
quorum; p. 4	A configuration of processing nodes capable of accomplishing the mission. A *quorum* is achieved if sufficient *healthy* nodes (*i.e.*, nodes that have not failed) remain capable of communicating among themselves. In graph-theoretic terms, a *quorum* is equivalent to a *connected subgraph* (or *component*) *induced* by deleting, from the *graph* corresponding to the original network, *vertices* and *edges* corresponding to faulty *nodes* and *channels*. The basic idea of a *quorum* as a *connected induced subgraph* may be varied by imposing constraints (*e.g.*, we could insist the *quorum* be a *tree* [30] or an array [36]), or by relaxing constraints (*e.g.*, we may *almost surely* have *connectivity* among *almost all* good *nodes* [39]). Moore and Shannon were evidently first to use *quorum* in the context of fault tolerance [48]. In the 1980's, Digital Equipment Corporation applied *quorum* to characterize its VAXclusters
span, spanning; p. 7	A set of *graphs* which covers the *vertices* of a *graph* G is said to *span* G
star; p. 10	*Tree* with but one *interior* vertex
subgraph: induced, monotonic; pp. 7, 14	*Subgraph*: delete any combination of *edges* or *vertices*. To *induce a subgraph*, delete *vertices*, along with all *edges* that impinge on each *vertex* deleted. Also see *edge-induced*, p. 14.
topology; p. 4	In the context of networks, the popularly entrenched *topology* tends to be *mis*-used in place of the arguably more accurate *adjacency* (which see). Properly speaking, a *topology* is a set of open subsets of a set S, such that a) S is open; b) the empty set is open; c) the intersection of any two open sets is open; d) the union of any number of open sets is open [53]. A tortured case for net-centric use of *topology* argues that, for the *topology* induced by the *edge distance space* of a *connected*, *unweighted graph*, the *open balls* of radius less than two imply the *adjacency*. Cautioning against mistaking *topology* for *adjacency*, for example: discussions of embeddings of *graphs* on surfaces with respect to bona fide *topological* properties, such as *genus* (a plane or sphere has *genus* zero; a toroid or donut has genus one [7])
transitive closure	In context: *factorizations* and workload *(di)graphs*, p. 8
tree; leaf, interior; p. 7	Connected *graph* without *cycles*. A *tree vertex* of *degree* one is a *leaf*, else it is *interior*

Table 2: Basic notation, cross-referenced and as used in this paper.

$\lceil x \rceil$; $\lfloor x \rfloor$; pp. 4, 5	*Ceiling* (least integer no less than x); *floor* (greatest integer no greater than x)
$\lvert X \rvert$; $\lvert P \rvert$; $\lvert \boldsymbol{A} \rvert$; pp. 9, 17	*Cardinality* of X; *pathlength* of P; *primality* (*i.e.*, *cardinality* of) *factorization* $\boldsymbol{A}$ (p. 7)
$\lvert x, y \rvert$; $\lvert x, y \rvert_G$; $\lvert x, y \rvert_{\mathbf{H}}$; Thm 1	*Distance* between x and y. *Edge*: $\lvert x, y \rvert$; in *graph* G $\lvert x, y \rvert_G$; *Hamming*: $\lvert x, y \rvert_{\mathbf{H}}$ (p. 13)
$O(g(n))$; $\Omega(g(n))$; pp. 8, 9	Set of functions no greater *resp.* no less than $c \cdot g(n)$, for $n > k$, constants c, k
$o(g(n))$; $\omega(g(n))$; pp. 9, 19	Set of functions $h(n)$ such that $\lim_{n \to \infty} h/g = 0$ *resp.* $\lim_{n \to \infty} g/h = 0$
$\Theta(g(n))$; p. 9	Exact order of magnitude: intersection of $O(g(n))$ and $\Omega(g(n))$
$\partial(x, i)$; $\partial(x)$; p. 19	*Closed ball in a graph*: set of vertices *edge distance* i from vertex x; *vertex neighbors* of x
e; $e_{\min}(n, f)$; Eqn. (2)	*Size* (number of edges) of a *graph*; minimum *size* of an $(f+1)$-*connected* graph
δ; f; p. 5, Eqn. (2)	*Degree*; nodal *fault tolerance*, one less than the *vertex connectivity*
G; V; E; p. 4	*Graph*, often one that represents a multi-processor or network; set of *vertices*; set of *edges*
$G_{\text{H-H}}(n, f)$; $G_{\text{LMF}}(n, f)$	$(f+1)$-*connected graphs*: Harary-Hayes (p. 5); extremal, with maximum *diameter* (p. 18)
$G_{\text{B}}(n, p, h)$; $G_{\text{H-L}}(n, p, h)$	Blough's *bipartite graph* (p. 19); LaForge's locally-spared *Hamiltonian cycle* (p. 9)
q; $\underline{q}$; p. 18	*Endpoints* of an *antipodal chord* in a Harary-Hayes *graph* $G_{\text{H-H}}(n, f)$
$K_j = K_j^1$; K_j^d, C_j^d; pp. 8, 13, 20	j-*vertex clique*, j-*hyperedge*; d-dimensional j-ary *K-cube*, *C-cube*
P; $P(x \ldots y)$; Table 1, p. 17	*Path*, perhaps directed; *path* with *endpoint*(s) x and y

An examination of the literature (*e.g.*, [2], [35]) reveals that both *bottom-up* and *top-down* are often weighed, or at least co-mingled, within the same exposition. It is nevertheless useful to contrast stock-in-trade viewpoints of (a) technologists with those of (b) system architects. Doing so offers the benefit of a trade-based forum for approaches to spacecraft avionics that achieve cost-effective fault tolerance, autonomy, and evolvability.

Consider, for example, a recent call by the United States Missile Defense Agency (MDA) for fault-tolerant spaceflight multi-processors that support *grid computing* [56]:

> *... To space systems designers, the performance of leading edge computing systems are often inaccessible, since these systems do not meet the requirements for reliable, failure-free life in a hostile (*i.e.*, radiation) environment. Currently, the most powerful rad-hard processor operates at 250 MIPS. In contrast, desktop computers are available that greatly outpace this capability. To compensate in part for this problem, adaptive computing approaches offer the possibility to blend reconfigurable computing resources into general purpose processors. The resulting processors can accelerate special types of computation (for example, finite impulse response filtering) by orders of magnitude. If such processors could be chained together in different network topologies, a rad-hard adaptive grid network would result, compensated in performance by built-in reconfigurable resources that can be tailored for mission-specific needs.*
>
> *For growth options, a distributed parallel processing approach is needed, since even the most powerful single processor has a limited performance level. New computational schemes, referred to as 'grid computing', appeal to a painlessly scalable network in which computing is increased in a manner analogous to the terrestrial power grid.*
>
> *This feature, in conjunction with other conventional fault tolerance approaches, would result in a flexible, robust computing architecture. Therefore, innovative solutions to distributed spacecraft processing are sought ... that combine radiation tolerance with adaptive computing in a network-centric scheme ...*

At heart, the preceding suggests dual-pronged integration of of *objectives* and *constraints*, which may in turn be profiled as

a) *Bottom-up*: rad-hard circuits, with attendant designs, manufacturing processes, and analyses.

b) *Top-down*: parallel grid computing.

In either case (a) or (b), moreover, we can combine approaches which might saliently be categorized as

i) *Broad*. *E.g.*, what commercial off-the-shelf solutions (COTS) are available to meet the goals of, or loosen constraints on, rad-hard net-centric grid computing?

ii) *Deep*. *E.g.*, what theoretical or practical problems can we solve, analytically or experimentally, that will enable rad-hard net-centric grid computing?

In a top-down (b) yet deep (ii) fashion, this paper substantially answers many of the challenges posed by the MDA call for spaceflight net-centric grid computing. Refining MDA's requirement for "processors chained together in different network topologies", our Theorem 5 – explained in Section 2 and proved in Sub-appendix A.6 – nails down necessary and sufficient conditions for the *existence* of an attractive class of network topologies, naturally modeled as *Hamming graphs*. Theorem 5 serves to fuel $\boldsymbol{A}_{\text{Label-Hamming}}$, a new, efficient algorithm that recognizes and labels a Hamming graph, or correctly declares it not Hamming. To warm up to our technical development, we present a whirlwind tour of related, graph-theoretic approaches pertaining to grid computing on fault-tolerant spaceflight multi-processors.

1.1. Top-Down Menu: From Sparse to Dense, a Tunable Range of Fault Tolerance and Connectivity

Addressing a general audience, [44] surveys a top-down, *broad* approach to MDA's call for adaptive topologies. Let us bolster *in-depth* reasons for such a top-down approach. Judging from historical precedents, such as [23], we can characterize one prevailing design habit as

Attempting to glean trends based on repeated simulations of variations on topologies and workloads (1)

While simulation is certainly valuable to the high-fidelity crafting of networks, abject application of method (1) suffers several drawbacks. The availability of network simulators (*e.g.*, *ns-2* [23] and OPNET [52]), along with the speed of modern computers on which to run simulations, seduces the architect to simulate first, think later. Succumbing to this temptation tends to shortchange critical thought, hence inhibit understanding of the fundamentals. For example:

What is the fewest number $e_{\min}(n, f)$ of point-to-point channels assuring *all* healthy nodes remain in a (connected) quorum, no matter how f faults are distributed? (2)

Topology vs. Adjacency — Before proceeding further, we should confess that this paper surrenders to the popularly entrenched *mis*-use of *topology* to mean "which nodes are connected to each other" (as opposed to the more proper, graph-theoretic "adjacency"). Tables 1 and 2 explain this, as well as other nuances of nomenclature and notation. With this clarification, and under a model of *uniform channel cost*, (2) asks for the rock-bottom price that we must pay for an n-node topology that tolerates f faults, in the worst case.

By *fault tolerant* we mean that the remaining $n-f$ nodes form a *quorum*: *i.e.*, between any two of the $n-f$ healthy nodes, there is a *path* traversing healthy nodes and channels. Harary [24] and Hayes [25] employ now-standard graph-theoretic arguments to derive the exact solution to (2):

$$e_{\min}(n, f) = \lceil n(f+1)/2 \rceil \quad \text{(Harary-Hayes Bound)} \qquad (3)$$

Why *graph-theoretic*? A *graph* G consists of *vertices* V, along with *edges* E, each of which joins two vertices. Graphs naturally model networks with faults: edges joining vertices correspond to point-to-point channels linking nodes. Under the graph-theoretic fault model set forth above, *no* technique – whether *ad hoc* or systematic – can best (3).

Deleting vertices or edges models faulty nodes or channels. A *quorum* comprises a graph *component*. Enriching our basic graph model, the *arcs* of *directed graphs* (or *digraphs*) model *one-way* (or *simplex*) channels. We may (and do) attach numerical *weights* to vertices, edges, or arcs. Weighted graphs capture, say, the information capacity of a channels, or power expended to maintain channels. Weights may also bracket the delay of packets traversing a node.

Published literature over the past fifty years suggests that graph theory offers more to a top-down, quantitative understanding of networks than any other discipline. It is therefore perhaps surprising that, over the past four years, *none* of the eight self-professed network architects whom The Right Stuff of Tahoe interviewed had even considered some formulation of (2), much less applied the solution (3). Ignorance of analytic fundamentals, such as the Harary-Hayes bound (3), deprives us of a great deal of predictive power for quantifying network behavior. By contrast, *intelligent application* of fundamentals, such as (3), greatly enhances our ability to quantitatively predict network behavior.

On the issue of scalability, for example, it appears that communities comprising both researchers and practitioners accord considerable importance to the following question.

When gauged by the total count of channels, how does the cost of worst-case fault tolerance scale as a function of the number n of nodes? (4)

Of course (4) is just (2) rewarmed, so the answer to (4) is given by the expression for $e_{\min}(n, f)$ in (3). To put a face on this formula, let the worst-case fault tolerance $f = f(n)$ scale as a function of the total number of nodes n (*cf.* [44] Fig. 13), and substitute $f(n)$ into the righthand side of (3).

If f is *constant* then the minimum channel cost $e_{\min}$ is a *linear* function of n. To achieve constant fault tolerance, that is, the minimum number of channels per node equals a constant $f+1$, when n and $f+1$ are not both odd. If both n and $f+1$ are odd, then one "stray" node maintains $f+2$ channels (again, a constant), while each of the remaining $n-1$ nodes maintain $f+1$ channels. We deem such a topology *sparse*: the number of channels per node is constant (or tightly bounded by constants); the worst-case fault tolerance equals a constant f.

At the other extreme, suppose that $f(n)$ grows as a constant *fraction* of the total number n of nodes, or approximately so. More precisely, suppose $f+1 = p \cdot n$, with $0 < p < 1$ constant. Then, applying the Harary-Hayes formula (3), the minimum channel cost $e_{\min}$ scales from a linear function of n, as recounted in the preceding paragraph, to a *quadratic* function of n. We deem such a topology *dense*: the count of channels per node is the number n of nodes times a constant p (or, for some node in the topology, $p \cdot n+1$ channels); the worst-case fault tolerance $f = p \cdot n - 1$ scales linearly with n.

The examples above peg endpoints in the *range* of tunable fault tolerance. Let us recast these results using the vocabulary of graph theory. The *degree* δ of a vertex is the number of edges to which it belongs. A graph G is *connected* if every pair of vertices in G comprise the endpoints of at least one *path* in G. The *vertex* (resp. *edge*) *connectivity* of G is the minimum number of vertices (*resp.* edges) whose deletion results in a disconnected graph, or a lone vertex. Equivalently, the vertex connectivity $f+1$ equals the minimum, over all vertex pairs, of the maximum number of *interior-disjoint* (IDJ) paths, starting and ending at the same endpoints, but otherwise not intersecting ([10] Thms 5.10, 5.11). Thus, a graph whose vertex connectivity equals $f+1$ models a network whose worst-case tolerance to faulty nodes equals f. For tunable multi-processors, therefore, both the fault tolerance f *and* the connectivity $f+1$ range from sparse to dense.

1.2. The Mathematics of Connectivity: From MTAD, to Fault Tolerance and Cost, to Latency and Throughput

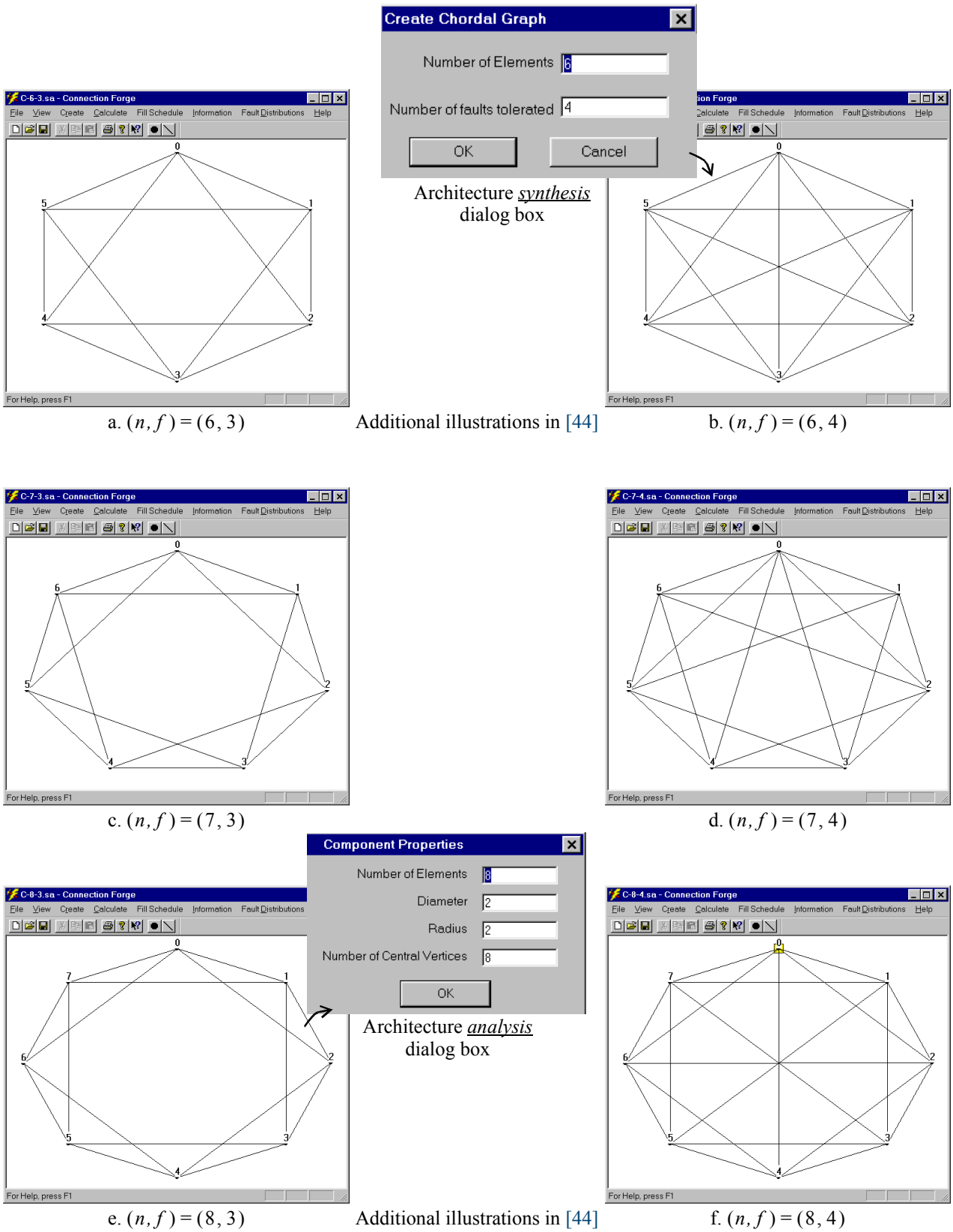
As a segue to Section 2, let us bolster the case for graph-theoretic tuning of multi-processor topologies that respect *multiple* objectives and constraints. Examples in Section 1.1 answer questions, such as (2) and (4), about how minimum channel cost scales as a function of both the worst-case tolerance to f faulty nodes, and the number n of network nodes. A stronger formulation of (2) asks for an f-tolerant topology that actually minimizes the number of channels.

Rising to this more stringent challenge, *regular chordal graphs* constructively achieve the value of $e_{\min}$ in (3) for least-cost fault tolerance. We denote such *Harary-Hayes graphs* by $G_{\text{H-H}}(n, f)$, after the researchers who introduced them. The software whose screen shots appear in Figure 1 synthesizes $G_{\text{H-H}}(n, f)$ using algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$, expressed below in the language of graph theory.

Synthesis and Labeling Algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$
% Input: connectivity $f+1 \geq 2$, vertex count $n > f+1$
% Output: Harary-Hayes graph $G_{\text{H-H}}(n, f)$
1) If $0 = (f+1) \bmod 2$ % Even connectivity
2) Then Label n nodes from 0 to $n-1$
3) For $i = 0$ to $n-1$
4) For $k = 1$ to $\frac{1}{2} \cdot (f+1)$
5) Add edge $(i, [i+k] \bmod n)$
6) Output the set of vertices and edges as $G_{\text{H-H}}(n, f)$
7) Else Starting with $G_{\text{H-H}}(n, f-1)$ % Odd connectivity
8) For $i = 0$ to $\lfloor \frac{1}{2} \cdot (n-1) \rfloor$ % Antipodal chords
9) Add edge $(i, [i + \lceil \frac{1}{2} \cdot n \rceil] \bmod n)$
10) Output the set of vertices and edges as $G_{\text{H-H}}(n, f)$

For the sake of clarity we adopt the convention of [42] for distinguishing congruences $\equiv$ and modulo equalities $=$. We take the *principal value* of $i \bmod j$ as the unique integer h such that $i = qj + h$, for some integer q and $0 \leq h \leq j-1$. We use the equal sign to emphasize *evaluation* of the righthand side to its principal value on the lefthand side. Thus, $5 \equiv 16 \bmod 11$, $5 = 16 \bmod 11$, and $-6 \equiv 16 \bmod 11$ comply with this convention, but $-6 = 16 \bmod 11$ (implying a principal value outside the interval [0, 10]) does not.

Despite optimizing the tradeoff between channel cost and fault tolerance, Harary-Hayes topologies exhibit abysmally high packet *latency*. That is, and as Section 2.2 spells out, the *radius* or *diameter* of a Harary-Hayes graph is very nearly as bad as it gets, even in the absence of faults. Provably bounded, minimum or near-minimum latency is particularly crucial to avionics multi-processors and realtime networks.



a. $(n, f) = (6, 3)$ Additional illustrations in [44] b. $(n, f) = (6, 4)$

c. $(n, f) = (7, 3)$ d. $(n, f) = (7, 4)$

e. $(n, f) = (8, 3)$ Additional illustrations in [44] f. $(n, f) = (8, 4)$

Figure 1: Tunable topologies. The Right Stuff of Tahoe's Connection Foundry™ design software synthesizes them by applying a *catalog* of theorems and algorithms. Above: 3 and 4 fault-tolerant Harary-Hayes graphs $G_{\text{H-H}}$ on 6, 7, and 8 nodes. For any count n of nodes, Harary-Hayes graphs optimize the tradeoff between overall channel network cost and [44] worst-case fault tolerance f. If our objective is to minimize *latency*, on the other hand, then Harary-Hayes graphs are about as far from optimal (*cf.* Section 2.2) as we can get: the radius and diameter are very nearly the worst possible!

In a recent request for proposals, for example, the United States Navy [57] issued a call for W Band avionics based on

> *A complete mathematical foundation … for end-to-end operation that provides predictable, stable performance. The performance characteristics of the whole network must be guaranteed and stable within the constraints of temporal accuracy, stability and high utilization. Graceful degradation is required and must be based on a mathematical foundation … that emphasizes testability and predictability.*

Sections 2.1 through 2.3 synopsize intricacies of topologies that simultaneously satisfy the Navy's call for graceful degradation and low, predictable latency. To this end, Figure 9 of [44] depicts how we would benefit from an *automated* knowledge catalog of theorems and algorithms. Recognizing the interdisciplinary nature of this knowledge, we will henceforth refer to it as the *mathematics of connectivity*. We will have attained a new level in the state-of-the-art when software can comprehensively and reliably apply the mathematics of connectivity to the synthesis of tunable topologies. Figure 1 illustrates how such topologies may be synthesized at *design time*. Alternatively, they may be synthesized in *realtime*, via algorithms distilled from design software, and embedded in the operational nodes. The nodes may be coupled by way of, say, VCSELs [43], or (*cf.* [44] Sec. 2.1) they may comprise a MANET, even more loosely coupled via radio-frequency (RF) channels.

Complementing the nascent work of [40], this paper contributes to a foundation for the synthesis of tunable, multi-processor topologies. The mathematics of connectivity is at once in need of analytic solutions to pressing problems, yet ripe with solutions for practical application. As to the former, Section 2 and Appendix A.6 answer heretofore open questions about Hamming graphs [31], study of which stems from objectives for optimizing fault tolerance, throughput, cost, and latency. The remainder of this section highlights the mathematics of connectivity with results that are ripe for practical application to the synthesis of tunable topologies.

Weighted Graphs: Directed vs. Undirected, Mutual Test and Diagnosis (MTAD) — To establish worst-case diagnosability with minimum count of Byzantine (also known as "PMC") tests, Preparata, Metze, and Chien [51] employ MTAD architectures which are, in effect, *directed* forms of Harary-Hayes chordal graphs: each arc corresponds to a test, with direction denoting which node performs the test, and which node is tested. A *syndrome digraph* signifies the test outcomes, pass or fail, via binary weights on each test arc.

In the interest of attribution, we should add that the seminal PMC paper [51] precedes the contribution of Hayes [25], which in fact cites [51]. Curiously, neither [25] nor [51] references Harary [24] as a proper predecessor. LaForge *et al.* [39] show how *undirected* graphs, quite different from the Harary-Hayes construction, minimize the cost of *probabilistic* MTAD under an *agreement graph* model of comparison. Essential to quorum configuration, fault diagnosis poses a significant challenge in its own right. As but one starting point to the expansive literature on fault diagnosis, LaForge and Korver [40] survey MTAD in the context of avionics.

Covers, Packings, Factors, Factorizations, and Matchings — Beyond MTAD, four sets of results illustrate key concepts from the mathematics of connectivity, ripe for practical application on a serious scale. These concepts – as well as the results which illustrate them – are germane to our exposition, and so warrant a modicum of attention to detail.

A *cover* of set S is a set of sets whose union contains S. A classic NP-Complete problem, aptly known as SET COVERING [21], takes as input an integer k, along with a set $\boldsymbol{Q}$ of subsets of S, and asks for a cover $\boldsymbol{Q}_k$ of S using at most k of the members of $\boldsymbol{Q}$. Any algorithm which correctly solves SET COVERING must also be capable of deciding the case where no cover exists. One variant, predictably deemed MINIMUM SET COVERING, asks for either a cover $\boldsymbol{Q}_{\min}$ of S by *fewest* members of $\boldsymbol{Q}$, or a (correct) declaration that no cover exists [11]. In a so-called *defect* (or *fractional* [46]) variation, we seek a collection of members of $\boldsymbol{Q}$ whose union maximizes the number of *elements* from S.

By contrast, a *packing* of set S is a set of sets whose union is contained in S. Packings may be constrained by *disjointness*, in any of several senses. Also known as *independence*, *pairwise* disjointness is strongest and perhaps most typical: no two members are allowed to intersect. An independent packing is known as a *matching*. Analogous to MINIMUM SET COVERING, for example, we can ask for a *maximum matching*; *i.e.*, a collection $\boldsymbol{R}_{\max}$ with the most number of independent members chosen from a set $\boldsymbol{R}$ of subsets of S. A packing of S is *complete* if the union of its members equals S. A complete matching constitutes a *partition* of S.

Applying these set-theoretic notions to the domain of graphs, each of S, $\boldsymbol{Q}$, and $\boldsymbol{R}$ generally reduces to some combination of vertices and edges. A set of graphs which covers the vertices of a graph G is said to *span* G. For example, not only does any connected graph possess a spanning tree, but any vertex x in a connected graph roots a spanning tree *that preserves the edge distance* from x [49].

To further illustrate how set packing specializes to graphs, let S be a digraph D, completely matched by a set $\boldsymbol{A} = \{A_0, \ldots A_{k-1}\}$ of subgraphs of D, pairwise arc-disjoint, but with each A_i spanning all vertices of D. Any such $\boldsymbol{A}$ is a *factorization* of D, and each A_i is a *factor* of D. Analogous definitions hold for undirected graphs ([8] p. 51). By definition, a factorization is complete with respect to D. If we relax the condition that A_i span D then A_i is a *weak factor* of D, and $\boldsymbol{A}$ is a *weak factorization* of D. If $\boldsymbol{A}$ need not cover D, then $\boldsymbol{A}$ is a *partial* (or *incomplete*) *factorization* of D. The *primality* of $\boldsymbol{A}$ is the number $|\boldsymbol{A}|$ of factors. If $|\boldsymbol{A}|$ is minimum then $\boldsymbol{A}$ is *prime*.

Classically, we seek to minimize primality $|\boldsymbol{A}|$, maximize the *size* of factors A_i. For example, if every A_i is an arc matching of D then $\boldsymbol{A}$ may be: i) a *one-factorization*; ii) a *weak* one-factorization (WOF), if some matching is incomplete; iii) a partial one-factorization, if $\boldsymbol{A}$ does not cover D. The qualifier "one" arises from every vertex in a complete matching having degree one. While you quite possibly have experience with algorithms for computing maximum (not necessarily complete) matchings ([8] Chap. 2, [20], [50] Sec. 10.2), you are perhaps less familiar with prime factorizations of graphs.

Clique-Factorized Optimization of MANET Throughput — To put a practical face on covering, packing and factorization, let us describe how minimizing one-factorizations can minimize *network schedules*. A *workload digraph* D_n specifies the source and destination of messages between nodes, but does not say *how* to effect delivery, whose job is in turn prescribed by a *network schedule*. Our objective here is to compute a minimum factorization $\boldsymbol{Q}$ of a *scheduling quorum* digraph Q, such that the transitive closure of Q covers D_n. Each factor Q_i models constraints, such as mutual-exclusion (MUTEX) access to an antenna. The *transitive closure* is just the digraph obtained by drawing an arc from vertex u to vertex v whenever there is a directed path from u to v. Thus, quorum connectivity is both necessary and sufficient for every packet to make its way to its destination, according to the workload specified by D_n. Since each factor Q_i occupies a timeslot, we seek to minimize the primality of $\boldsymbol{Q}$.

In the case of MANETs with directional antennas (or free-space lasers), an n-vertex (directed) *clique* models the workload of the following challenge problem.

Given an array of n nodes, uniformly packed in a rectangular grid, what is the fewest number of discrete timeslots to deliver a unique bit packet, of fixed length, for every possible source-destination pair? (5)

Formulation (5) above is a variation on benchmarks devised as part of the United States Defense Department's effort for coordinating the emerging technologies of software defined radios (SDRs) [16]. In 2004, The Air Force Research Laboratory posed (5) to five contractors, including The Right Stuff of Tahoe. The barebones version of (5) postulates a single antenna at each node, which either transmits at constant bit rate to at most one node, or receives (at the same constant bit rate) from at most one node. Such a MUTEX constrains factors of a scheduling quorum as *arc matchings*, in the case of half-duplex channels, or *edge matchings*, when channels are full-duplex. To summarize [32]: the transitive closure of any such minimum factorization *is* the scheduling quorum. Absent noise and interference, an optimum schedule delivers all messages in a number of timeslots equal to the primality of a weak one-factorization (WOF) of an n-vertex *clique*:

([32] Thm 3) Let $T_{\min}$ be the fewest number of timeslots in any schedule satisfying the barebones version of (5). If each node communicates over at most one full-duplex channel at a time, then $T_{\min} = (n - 1 + n) \bmod 2$. If each channel is half-duplex then $T_{\min} = 2[(n - 1 + n) \bmod 2]$ (6)

Our work on (5) uncovered a surprising gap in the mathematics of connectivity. While factorization of cliques into *Hamiltonian paths* and *cycles* is a standard result ([8] p. 94), *one-factorization* of cliques was, prior to [32], apparently an open question. To constructively prove (6), we were faced with filling this gap with original theorems and algorithms that synthesize factorizations whose primality matches that of a (simple) lower bound. With upper or lower bound constructively so derived, such an approach epitomizes what graph theorists have come to call the *extremal* method [8]. Let us flesh out some of the subtleties that led to the provably correct algorithm $\boldsymbol{A}_{\text{Clique-WOF}}$ [32], thus *constructively* establishing the upper bound for (6).

For example, we might be tempted to conjecture that iteratively applying a deterministic algorithm for finding a maximum matching will correctly one-factorize K_n into fewest matchings. At the outset, that is, we might surmise that the regularity of cliques has a good chance of assuring correctness of the following.

Factorization Algorithm $\boldsymbol{A}_{\text{Maximum-Matching}}(n)$
% Input: Number of nodes n
% *Alleged* Output: Minimum weak one-factorization of K_n,
1) $i = 0$ % Initialize number of matchings
2) While there remain unmarked edges of K_n
3) Let Q_i be a maximum matching of unmarked edges
4) Mark the edges of Q_i
5) Output Q_i
6) $i = i + 1$
7) Output i % Total number of matchings

Replacing line 3 of $\boldsymbol{A}_{\text{Maximum-Matching}}$ with an invocation to a maximal matching heuristic might lead us to suspect the alleged minimality of the factorization output by the algorithm. For example, a *greedy* heuristic iteratively adds independent edges to a matching, without backtrack, until no more edges can be added [11]; *i.e.*, the greedy matching is maxi*mal*. However, counterexamples abound for maximal matchings that are not maximum ([32] Fig 9). To achieve bound (6), moreover, every weak-one factor must match $n - n \bmod 2$ vertices; *i.e.*, each matching is maximum. This bears out suspicions that a maximal matching heuristic may not minimize the number of matchings. However, it is perhaps surprising that $\boldsymbol{A}_{\text{Maximum-Matching}}$, with maximum matching at line 3, can fail to factor a clique into the fewest matchings. By contrast with the provably correct $\boldsymbol{A}_{\text{Clique-WOF}}$ of [32], $\boldsymbol{A}_{\text{Maximum-Matching}}$ just doesn't nail the minimum number of factors. That said, we have *experimentally* calibrated the correctness of a greedy matching at line 3.

Suppose we use a *heap* to implement a *priority queue* [13] of vertices, sorted by increasing number of unmatched neighbors. Then greedy matching is both faster $O(n \log n)$ and simpler than the most efficient general algorithm $O(n^3)$ known ([50] Sec. 10.2) for graph matching. In trials on 16, 25, and 100 nodes, for example, greedy matching at line 3 gives schedules whose count of discrete timeslots runs 10%, 6%, and 2.5% over the respective optima, as given by (6).

Even for wireless communications, (6) is a good approximation, as long as channels are nonoverlapping or (as in the some cases of free-space VCSEL laser light) pencil beams. Otherwise, electromagnetic interference constrains the solution to (5). Figures 3 and 4 of [44] illustrate software which effects RF-feasible schedules based on clique factorization. Even here, (6) is handy for telling us the best possible schedule length. Using 2.4 GHz directional antennas with 3dB rolloff at 12.5° off boresight, for example, a 4×4 instance of (5) is feasible in 30 timeslots, the fewest possible predicted by (6). However, trends for scaling are not quite so encouraging: a 10×10 schedule completes in 518 timeslots, or approximately 160% on top of the optimum count 198 of timeslots predicted by (6). For reasons such as this, our study [32] reinforces the recommendation of [28]: frequency division into many nonoverlapping bands is highly preferable to time-division multiplexing on a single wide band.

Viterbi-Lookahead Factorization Enables Switch Fabric Throughput — Switch fabrics route traffic from different input channels (say, p of them) to output channels (say, q of them). For high-speed Internet trunks, including fiber optic longlines, switch fabrics reside in dedicated routing nodes. With either VCSEL-enabled multi-processors or RF-coupled MANETs, the communications layer of a general purpose node likely houses a switch fabric. The clocked duty cycle of a switch fabric entails i) examining packets from the p input buffers, or *ingresses*; ii) determining into which output buffer, also known as an *egress*, to place each packet; iii) transferring packets to their respective egresses.

Switch fabric designers seek to satisfy multiple objectives [26], some of which may conflict (*e.g.*, so-called best-effort fairness versus prioritized quality of service). By and large, however, the prevalent objective [18] is to maximize packet throughput across the fabric, in this case synergistic with reducing *latency*. We can represent the delivery workload by arcs flowing from input vertices to output vertices. If more than one packet from ingress x has been assigned an egress, then more than one arc emanates from input vertex x. If more than one packet has been assigned to egress y, then more than one arc impinges on output vertex y. In other words, the workload models as a $p \times q$ bipartite directed multi-graph $D_{p,\,q}$. If, as for some switching policies, we wish to preserve information about packets, (such as arrival time) then we can either encode that information as weights on arcs, or replace the tail of each arc with a unique vertex per packet. Note that, while the workload *clique* of challenge (5) is predictable, the workload represented by $D_{p,\,q}$ *varies*, perhaps unpredictably, with network workload.

In general, each egress outputs at most one packet in any timeslot. Within respect to this MUTEX, we can therefore pose our *throughput objective* as minimizing a weak one-factorization of $D_{p,\,q}$. Such a *lookahead* approach is reminiscent of Viterbi algorithms [19], but departs from mainstay efforts of investigators to (re)maximize a *single matching* of $D_{p,\,q}$, based on the state of the switch fabric at any timeslot. The current crop of algorithms [22] appears to ignore the impact on subsequent matchings. By contrast, we advocate one-factorizing $D_{p,\,q}$ into matchings, the i^{th} of which forecasts parallel packet transfer i timeslots later. Minimum factorization makes more comprehensive use of state information than does one-shot maximum matching. Incidentally, we can compute a maximum bipartite matching in time $O(n|E|)$ ([13] Sec. 27.3); this trumps the fastest known algorithm $O(n^3)$ for matchings of general graphs ([50] Sec. 10.2) when $|E| \in o(n^2)$. The literature suggests that investigators of switch fabrics may not all be aware of this fact.

Encouraged by our results (6) for factorization of cliques (5), we suggest that maximal (perhaps greedy) matchings would suffice to *efficiently* smooth jitter, at the same time approximating optimum schedules in offline cases where all packet egresses are known. The catch: although a matching predictably removes arcs in the active timeslot, new packets *add* an unknown distribution of arcs to $D_{p,\,q}$, over many timeslots. Quantifying the costs and benefits of switch fabric factorization appears to be an open problem, which we respectfully submit to researchers in the mathematics of connectivity.

Maximum Test Concurrency and Throughput for MTAD Syndromes — As surveyed in the lower lefthand column on page 7, results for mutual test and diagnosis (MTAD) tell us how to tune topologies to the minimum number of pairwise tests that a network or multi-processor must execute in order to diagnose (hence excise) faults. Aside from the matter of a distributed diagnosis algorithm, which *resolves* a syndrome of test outcomes by identifying faulty nodes, how can we schedule tests to run in minimum time? Reflecting established techniques for applying and measuring localized tests (*e.g.*, linear feedback shift registers, or LFSRs), we stipulate that a node tests, or is tested by, at most one other node at any given time. Such a test MUTEX calls for (weak) one-factorization of a test digraph D, akin to minimum-length schedules satisfying workloads for MANETs (5) and switch fabrics. But first, let's check that the number of tests in fact warrants the effort of minimum length scheduling.

LaForge and Korver [41] underscore a remarkable coincidence: the test cost of MTAD is essentially the same as the channel cost of fault tolerance, as noted on page 5, and as plotted in Figures 7, 13a, and 13b of [44]. Invoking order-of-magnitude notation (*cf.* Table 2), let us summarize how these costs pertain to n-node tunable topologies. To diagnose or tolerate faults, the minimum test or edge cost is: a) $\Theta(n)$, for a constant *number* of faults, and in the *worst case*; b) $\Theta(n^2)$, when the worst-case fault tolerance scales as a constant fraction of n [51]; c) $\Theta(n \log n)$, to *almost surely* diagnose or tolerate a constant fraction of independent, identically distributed (IID) faults, and where nodes have bounded degree [6], [54]; d) $\Theta(n)$, under the same conditions as probabilistic case (c), but *where we are allowed to misdiagnose an arbitrarily small fraction of healthy nodes* (hence permit them to be disconnected from the quorum [39]).

Since the number of tests is so substantial – from $\Theta(n)$, to $\Theta(n \log n)$, to $\Theta(n^2)$ – it is indeed worth our effort to schedule D into fewest timeslots. The test MUTEX mentioned above implies that any such schedule is at least as long as the degree δ of any vertex of D. Therefore, schedule length is at least $\lceil \delta_{\text{avg}} \rceil$, with the total number of tests $\frac{1}{2} \cdot n \cdot \delta_{\text{avg}}(n)$ scaling as described in the preceding paragraph. Fortunately, we can one-factorize D into the minimum number $\lceil \delta_{\text{avg}} \rceil$ of matchings, for at least two parameterizable classes of test digraphs: i) D is a *directed* Harary-Hayes graph $G_{\text{H-H}}(n, f)$; ii) D comprises a *directed Hamiltonian cycle* on the *blocks* of *locally-spared* topologies $G_{\text{H-L}}(n, p, h)$, with $f+1$ vertices in each block (original analysis, for VLSI arrays, in [37] and [38]; details in Sub-appendix A.4; Figure 8 of [44] illustrates for $n = 4$, $h = 2$). The fewest timeslots to compute a syndrome is a) $\Theta(1)$, for a constant *number* of faults, and in the worst case; b) $\Theta(n)$, when the worst-case fault count scales as a constant fraction of n; c) $\Theta(\log n)$, to *almost surely* diagnose or tolerate a constant fraction of IID faults, and d) $\Theta(1)$, for almost sure diagnosis or tolerance, and where we are allowed to misdiagnose as faulty an arbitrarily small fraction of healthy nodes. Results (a) – (d) might impel us to prefer MTAD's speed over that $\Omega(n)$ of conventional serial testing, where the latter includes techniques akin to boundary scan [47]. It remains for researchers in the mathematics of connectivity to expand the catalog of one-factorizable test digraphs, and to put this catalog into practice.

From Workload One-Factorizations, to Matching Approximations for Covers, to The Problem of Zarankiewicz — The preceding subsections underscore how applying key notions from the mathematics of connectivity can maximize the throughput of MANETs, switch fabrics, and mutual test and diagnosis. We seek to decompose a graph into fewest factors – *matchings*, for the cases in point. Recalling our overview on page 7, matchings are but specialized *packings*. Let us highlight a potent, complementary notion: *covers*.

Kuo and Fuchs [29] put a fine point on why the term *fault cover* is so apt. The authors reduce to BIPARTITE VERTEX COVER the problem of assigning a combination of dedicated spare rows and spare columns that replace (*i.e.*, cover) all the faulty processing elements in a two-dimensional array. Network architects may find it counterintuitive that, in this case, it makes sense to model processing elements as graph edges, rather than as vertices. Corresponding to the p rows and q columns containing faults, let B denote a $p \times q$ bipartite graph. Join vertex i, from the left vertex set of B, with vertex j, from the right, if and only if array element *(i, j)* is faulty. We can switch in h fault-free spare rows and k fault-free spare columns (hence achieving a fault-free array) if and only if we can cover all the edges of B, using at most h vertices from its left vertex set, and k vertices from B's right vertex set. Although BIPARTITE VERTEX COVER is NP-complete, matching can help out, even here. By a result generally attributed to König [7], the size of a maximum bipartite matching equals the order of a minimum bipartite cover. Thus, choosing each vertex from a maximum matching gives a bipartite vertex cover whose order is at most twice the minimum. Adapting ideas from matching theory, LaForge derives a crossbar switch that optimizes the layout area of arrays spared by dedicated rows and columns ([33] Sec. 3.2).

Solutions that help us tune multi-processors are by no means limited either to matchings of vertices by edges, or to covers of edges by vertices. For example, LaForge [33] establishes conditions on s, t, p, and q for a matching of *stars* (*i.e.*, trees containing but one interior vertex) to cover all $s \times t$ bipartite cliques in a $p \times q$ complete bipartite graph. This yields a König-type result that tells us when a *maximum* matching of stars *is* a *minimum* cover of cliques. The practical upshot: how to tune the redundancy of fault tolerant arrays (*homogeneously* spared, in contrast with *dedicated* sparing cited in the preceding paragraph). This *application*-motivated result also exactly solves half of all instances of a longstanding open question in graph *theory*: the *Problem of Zarankiewicz*.

As another example where we match more than just edges or arcs, consider MANET challenge (5), this time with more capable antennas. *I.e.*, a node can simultaneously communicate with any number of nodes. To avoid self-interference, no node can transmit and receive at the same time. The attendant *directed star* MUTEX implies (as things turn out [32] Thm 2) that a prime factorization is a sequence of $T_{\min}$ edge matchings of *bipartite clique* matchings of K_n:

$$\lceil \log_2 n \rceil \leq T_{\min} \leq 2\lceil \log_2 n \rceil \quad \text{(Matching of matchings)} \quad (7)$$

Exponential improvement over (6) effectively relegates to footnote status the factor-of-two gap between the lower bound of (7), and the constructively obtained upper bound.

1.3 Recap of Section 1: Look It Up, or Figure It Out?

With specificity, we have illustrated how the mathematics of connectivity is ripe with solutions for practical application: Harary-Hayes graphs (fault tolerance at minimum channel cost), MTAD (diagnosability with fewest tests), switch fabric workload matchings (instantaneous maximum throughput), and arrays with dedicated row-column sparing (vertex covering) all fortify the importance of *cataloging* established results from the mathematics of connectivity. That said, it would be unrealistic to presume that already overtaxed engineers would enthusiastically embrace such a catalog, and make it their priority to digest and apply results from it.

We should therefore embody such results as an active, software-driven catalog (*cf.* Figure 1, [44] Figs. 3 and 4). From mission planning to operations, software that realizes tunable spaceflight multi-processors will intelligently apply the mathematics of connectivity. Recalling the MDA and Navy solicitations quoted on pages 4 and 7, the compute power of individual spaceflight processors will continue to lag that of their earth-based counterparts, and *by years*. Therefore, synthesis and analysis of topologies – from sparse to dense – will prove particularly beneficial for *spaceflight* multi-processors tuned across a range of optima.

New technologies present new opportunities for breakthroughs in how we connect multi-processors. For example, vertical cavity surface emitting lasers (VCSELs) obviate the tyranny of planar or limited-layer interconnect. To put this in perspective [14], the average degree δ_{avg} of a vertex in a planar graph is less than 6. Thus, with planar or limited-layer technology (*cf.* [28] Fig. 10), the number of channels per node is $\Theta(1)$. This severely hampers our ability to tap the full range of tunable topologies, whose degrees can (and should, recalling discussions on pages 5 and 9) be capable of scaling with the number n of nodes. For example, the best layout we know for an n-node binary hypercube costs $O(n^2)$ area, $O(n)$ wirelength [37]. With VCSELs, we estimate that we will be able to reduce the area of binary hypercubes to between $\Omega(n)$ and $O(n \log n)$, with wirelength $\Theta(1)$.

From sparse to dense, tunable fault tolerance and connectivity *are* within our reach. Intelligent tuning of topologies demands that we reverse Rent's Rule [12]. Instead of accepting cell, chip, or board pincount as an (empirical) power of circuit area or part count, we derive the number of channels per node, based on top-down tradeoffs among fault tolerance, throughput, latency, and cost (*e.g.*, power). Rather than plunge into second-order considerations – such as chip or board level routing – we focus on top-down synthesis of preferred topologies, according priority only to a handful of constraints that make the most difference.

Our examples also illustrate how the mathematics of connectivity is in need of solutions to pressing problems: clique factorization (MANET throughput), switch fabric workload factorization (maximum throughput in the long run), and the Problem of Zarankiewicz (homogeneous sparing of arrays) all underscore how we need not rely on pure theory to serve up our answers. As Section 2 unfolds, we can and will contribute fresh results to the mathematics of connectivity.

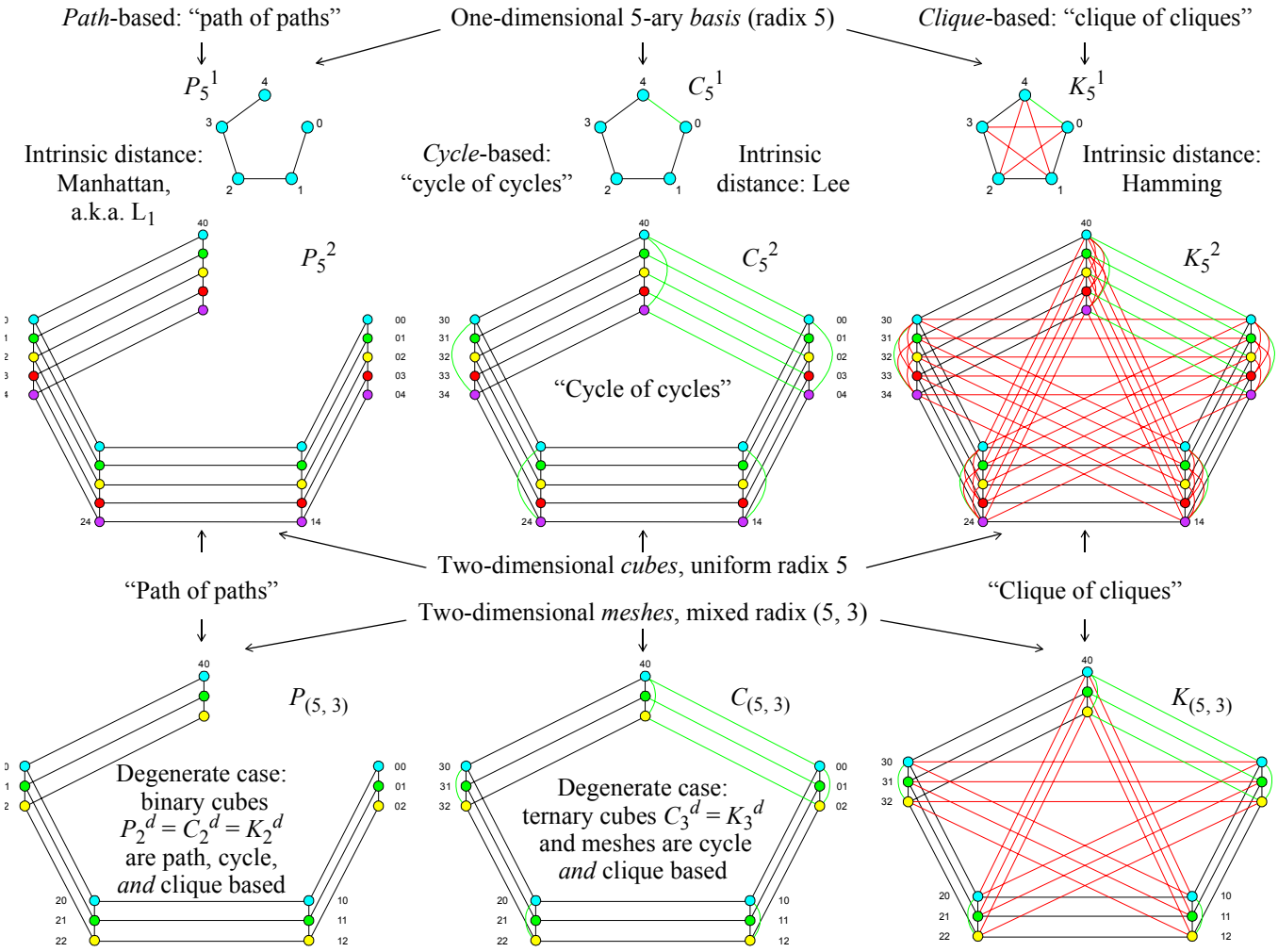
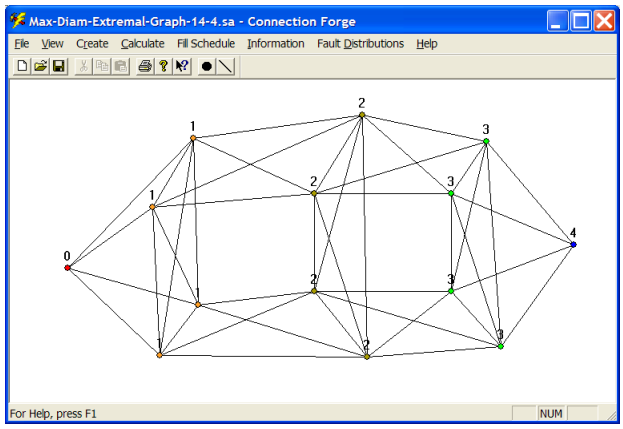


Figure 2: Hypercubes and meshes, dimension and radix. Also see Figure 9 of [44].



a. $(n, f) = (14, 3)$

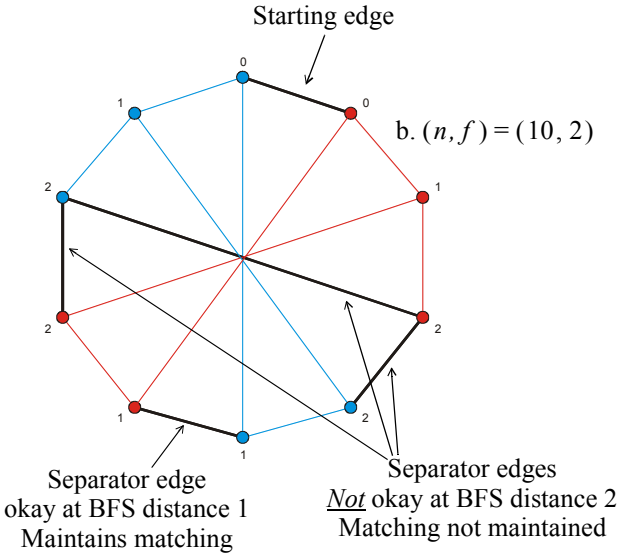


Figure 3: a) Extremal graph $G_{\text{LMF}}(n, f)$ of Theorem 2. b) Breadth-first search of Algorithm $\boldsymbol{A}_{\text{Label-Hamming}}$ exposes how the Harary-Hayes graph $G_{\text{H-H}}(2[2q+1], 2)$ of Theorem 4 fails the matching conditions of Theorem 5.

2. Theorem and Algorithm: Hamming Graphs Factorize As Hyperseparators

2.1. Multivariate Feasible Regions Serve to Tune Priorities

To date, the mathematics of connectivity has gravitated toward *univariate* optimization. We illustrate with two additions to the catalog proffered by Section 1.2. The textbook solutions below exemplify both the use of *weighted graphs (cf.* pp. 5, 7, 9), and the applicability of *trees* (*cf.* pp. 7, 10).

Euclidean Spanning Trees Minimize RF Network Power — A tree that spans an edge-weighted graph is minimum if the sum of the edge weights in no other spanning tree is less. When the weights equal the Euclidean distance between nodes in the plane or space, then we have a *minimum Euclidean spanning tree* (MEST). Over a wide range of models for radio frequency (RF) path loss, the Shannon channel capacity falls off logarithmically with the Euclidean distance [32]. Therefore: i) to minimize aggregate *power*, subject to ii) constant bit rate in iii) the fewest channels (zero fault tolerance) that iv) assure a quorum, we v) tune our topology to be an MEST. Using $\Theta(n \log n)$ Voronoi algorithms from computational geometry [14], this is both straightforward and (one would hope) familiar to designers of RF networks.

Code-Theoretic Huffman Trees Minimize Expected Latency — To further catalog results from the mathematics of connectivity (*cf.* p. 9), we draw – from classical *coding theory* – enablers for *probabilistically* tuning topologies. Assuming that the frequency distribution of messages is known (as it must be, with reasonable fidelity, else expectation cannot be computed), apply Huffman's prescription for minimizing expected message length ([13] Sec. 17.3). The attendant *Huffman tree* minimizes the expected edge distance from the root; *i.e.*, minimizes the average latency of messages emanating from a central node.

Like all trees, MESTs and Huffman trees fall woefully short of *tunable fault tolerance* and *throughput* – desirable properties which are often synergistic. That is, paying for more fault tolerance can (and should) serve up more throughput, and at no extra premium. To sharpen this point, suppose that channels have uniform cost and *information capacity*, with the latter our preferred *workload-independent* figure-of-merit for throughput. Recall from page 5 that we achieve f-fault tolerance via $f+1$ IDJ paths between two endpoints. By a factor of $f+1$, tolerance to f faults thus enhances the information capacity of a duplex channel. Regrettably, any tree is 1-connected. Thus, a tree not only imposes a *zero tolerance* fault policy, but chokes throughput down to the capacity of a single point-to-point channel. While FAT-trees can certainly augment throughput [32], they provide scant improvement in fault tolerance. Moreover, in many cases (*e.g.*, convoys along a long highway) wireless MESTs amount to a *path*, whose eccentricity (hence latency) is the worst possible.

To recap in the language of optimization: we seek *feasible regions* which enable us to tune, from sparse to dense, multiprocessor topologies for a) maximum connectivity (*i.e.*, fault tolerance and throughput); b) minimum eccentricity (*i.e.*, latency); and c) edge cost (*e.g.*, channel count, power).

2.2. Tunable Latency and The Moore Bound: Why Not to Use Harary-Hayes Graphs For Realtime Applications

As Sections 1.2 and 2.1 explain, Harary-Hayes graphs furnish a solid foundation for tuning fault tolerance, throughput, and channel count. Regrettably, and as mentioned on page 5 and by the caption to Figure 1, Harary-Hayes graphs suffer from *bloated latency*. To understand what we mean, consider this question: what are the maximum (*i.e.*, loosest) and minimum (*i.e.*, tightest) *radius* and *diameter*, over the *set* of $(f+1)$-connected graphs, including those with fewest edges (3)? Paralleling our scaling analyses on page 5 and page 9, and mindful that $\text{RAD} \le \text{DIAM} \le 2 \cdot \text{RAD}$ ([10] Thm 2.4), we use *eccentricity*, or ECC, to subsume RAD and DIAM.

Loosely speaking, Theorem 1 of Sub-appendix A.2 tells us that the eccentricities of $G_{\text{H-H}}(n, f)$ scale as $n/(f+1)$, for f even, and as $\frac{1}{2} \cdot n/f$, for f odd. These values are quite close to $1 + n/(f+1) \in \Theta(n/f)$, the diametric upper bound constructively achieved by Theorem 2's $G_{\text{LMF}}(n, f)$. By *orders of magnitude*, moreover, either $G_{\text{H-H}}(n, f)$ or $G_{\text{LMF}}(n, f)$ saddles us with eccentricity in excess of *the Moore Bound*:

$$\text{RAD} \ge \left\lceil \rho^{-}_{\text{Moore}}(n, f) \right\rceil \stackrel{\text{def}}{=} \left\lceil \log_f \left[\frac{n(f-1) + 2 + [n(f+1) \bmod 2]}{f + 1 + [n(f+1) \bmod 2]} \right] \right\rceil \quad (8)$$

For scaling objectives with tunable topologies, we find it handy to employ E. F. Moore's bound, derived in the 1950's, in the logarithmic formulation (8) of LaForge *et al* ([42] historical remarks, footnote 2). Notwithstanding (8)'s reliance on minimum *degree*, observe that the diametric upper bound of Sub-appendix A.3 loosens somewhat if we relax the $(f+1)$-connectedness condition of Theorem 2, and instead stipulate that every vertex have degree $f+1$ (*cf.* [8] p. 104, girth instead of diameter). More generally, $(f+1)$-connectedness implies every vertex has degree at least $f+1$, a fact exploited in Sub-appendix A.1. As with Theorem 2, constructions which effectively meet bound (8) do exist; however, *perfect* Moore graphs – *i.e.*, graphs whose diameter achieves equality in (8), with the ceiling symbols removed – are rare (*cf.* the Petersen graph, p. 15, [7] Sec. VIII).

To profile the tunable range of eccentricity, we proceed along the lines of (a) – (d) on page 9. If a) f is *constant* then the $\Theta(n/\log n)$ max-to-min ratio of eccentricities effectively relegates to footnote status the factor-of-two gap between $\text{DIAM}(G_{\text{H-H}})$ and $\text{DIAM}(G_{\text{LMF}})$, when n is even. The $\Theta(n/\log n)$ max-to-min ratio as well pertains for d), almost sure diagnosis and tolerance, where we may exclude an arbitrarily small fraction of healthy nodes from the quorum, and wherein vertex degree is minimized to a constant. For c), almost sure diagnosis or tolerance of a constant fraction of IID faults, and where each vertex of a minimum size graph has bounded degree $\Theta(\log n)$, the max-to-min decreases to $\Theta(n \log\log n / \log^2 n)$ (representative curves plotted in Figure 13c of [44]). b) Only when the worst-case diagnosis and tolerance jump to a constant fraction p, does the max-to-min eccentricity bottom out at $\Theta(1)$. Even here, however, we can trump the $\frac{1}{2} \cdot (1/p)$ eccentricity of $G_{\text{H-H}}(n, f)$ with diameter 2 or 3. To conserve page count, we forego details of such tightest possible, dense constructions.

Having spelled out reasons for relegating Harary-Hayes graphs to the bottom of our list of candidates for realtime topologies, we should add two caveats. i) An n-vertex cycle *is* $G_{\text{H-H}}(n, 1)$, which is, in fact, tightest possible (unique, in fact) over all minimum edge count 1-fault tolerant graphs.

ii) *Tight* may not be right. While contemporary applications tend to emphasize low or minimum latency, instances could conceivably arise where it makes sense to *maximize* the latency of $(f+1)$-connected topologies. As an example, albeit somewhat contrived, suppose that a node detects a computer virus as having penetrated quorum security. To impede viral propagation while the quorum collaborates to excise the intruder, nodes might *re*-tune from a topology having minimal eccentricity to one which is highly eccentric, such as that prescribed by a Harary-Hayes graph. Alternatively, nodes might re-tune to achieve $G_{\text{LMF}}(n, f)$, whose diameter achieves the absolute maximum among all $(f+1)$-connected graphs. For the latter, (19) reveals that we would pay a modest premium, as measured by count of edges, in the graph, or *channels*, in the network; *i.e.*, $1 < |E(G_{\text{LMF}})|/|E(G_{\text{H-H}})| \leq 1 + 2f/n + 1/(f+1)$. Absent a crystal ball that divines the priorities of our avionics clientele, it behooves us to *catalog* potentially applicable results from the mathematics of connectivity. Mission priorities should determine how to tune topologies, synthesized in turn by software that draws from a best-of-breed catalog.

2.3. Hamming Graphs: Low Latency and Cost, Fault Tolerance and Connectivity from Sparse to Dense

Preliminaries in hand, we can use the vocabulary of graph theory to pose our fundamental multivariate challenge:

What $(f+1)$-connected n-vertex graphs with fewest edges $\lceil n(f+1)/2 \rceil$ minimize the maximum a) radius or b) diameter of subgraphs (*i.e.*, *quorums*) induced by deleting up to f vertices? (9)

Consistent with DoD solicitations quoted on pages 4 and 7, but departing from a preponderance of research, (9) asks for *graceful degradation*: *i.e.*, minimax eccentricity *in the presence of faults*. Section 2.2 details how Harary-Hayes graphs exhibit eccentricity about as far from satisfying (9) as we could get … *even in the absence of faults*. So what graphs *do* satisfy (9)?

Many *Hamming graphs* satisfy (9). Of particular interest: *complete* Hamming graphs, synonymously known as *Hamming ideals*, *clique-based cubes*, or *K-cubes* ([44] Fig. 9). Why are theorems and algorithms about Hamming graphs important? Unless we understand the proper topologies for linking multi-computers, we cannot fully leverage interconnection technologies, such as VCSELs. The seven applications described in Sections 1.2 through 2.3 resound a recurrent theme: to properly tune multi-processor architectures, it behooves us to compile and apply a catalog of theorems and algorithms from the mathematics of connectivity. On par with Harary-Hayes graphs and MTAD, results about Hamming graphs lay rightful claim to an entire chapter in any such catalog. As a segue to this paper's contributions in depth, let us set the stage with a condensed yet proper introduction of Hamming graphs.

Hamming Graphs: Definitions and Notation — The *Hamming distance* $|x, y|_{\mathbf{H}}$ equals the number of positions where strings x and y, of the same length, differ. *Graph G* is *Hamming-labeled* if the edge distance between every two vertices equals the Hamming distance between the respective labels; *i.e.*, $|x, y|_{\mathbf{H}} = |x, y|_G, \forall\, x, y, \in V(G)$. A graph is *Hamming* if it can be Hamming labeled.

Strings x and y are *Gray-code adjacent* if $|x, y|_{\mathbf{H}} = 1$. Recursively construct a *d-dimensional Gray-coded j-ary K-cube* K_j^d as follows. K_j^0 is a lone vertex labeled with the null string. For K_j^d we i) make j copies of K_j^{d-1}; ii) join with an edge vertices u and v (from different copies of K_j^{d-1}) if and only if u and v have with identical labels; iii) prepend i to the label of each vertex of the i^{th} copy of K_j^{d-1}. Equivalently ([31] Cor. 2.3), we can Gray code a d-dimensional j-ary K-cube K_j^d by labeling $n_K = j^d$ vertices with successive d-digit values, from 0 to $j^d - 1$ inclusive. Join two vertices if and only if their labels are Gray-code adjacent. Figure 2 illustrates for $j = 5$ and $d = 1, 2$. Note that K_j^1 is just the clique K_j (also known as a *j-hyperedge*, or *j-edge*), whose vertices can be (and, for our purposes, have been) labeled from 0 to $j - 1$.

With labeling as prescribed in the preceding paragraph, the K-cube K_j^d constitutes the *unique*, *complete* Hamming graph with uniform *radix j* … complete in the sense that its vertices map one-to-one with the $n_K = j^d$ labels on d digits radix j, all while preserving a one-to-one mapping of edge distances between vertices, on one hand, and Hamming distances between the corresponding vertex labels, on the other ([42] Cor. 3.1). At $j = 2$ we obtain the familiar binary hypercube, and the Hamming distance specializes from the L_1 metric.

A Hamming labeling may have *mixed*, or *mesh*, radix $\boldsymbol{j} = (j_{d-1}, \ldots, j_0)$. In any label, that is, the i^{th} digit is an integer drawn from the interval $[0, j_i - 1]$, but j_i need not equal j_k. Without loss of generality, and unless specified otherwise, we shall consider radices to be *majorized*: $k \geq i$ implies $j_k \geq j_i$. When the $n_K = \prod_{0 \leq i \leq d-1} j_i$ labels radix $\boldsymbol{j}$ map one-to-one onto the vertices, then the Hamming graph is a *K-mesh* $K_{\boldsymbol{j}}^d$, *complete* with respect to the radix vector subscripted in boldface. We may drop the superscript when the dimension d is apparent or not pertinent. Alternatively, and as Figure 2 illustrates, we may explicate the mesh radix, *e.g.*, $K_{(5,3)}$, with dimension 2 implicit. Where all d digits of $\boldsymbol{j}$ range from 0 to $j - 1$, we may replace the boldface vector $\boldsymbol{j}$ with j; in this case the K-mesh $K_{\boldsymbol{j}}^d$ specializes to the K-cube K_j^d. Akin to our prescription for cubes, we Gray code a d-dimensional $\boldsymbol{j}$-ary K-mesh $K_{\boldsymbol{j}}^d$ by labeling n_K vertices with successive d-digit values, from 0 to one less than $n_K = \prod_{0 \leq i \leq d-1} j_i$. Join two vertices whenever their labels are Gray-code adjacent. Mindful that radices be majorized, note that $K_{\boldsymbol{j}}^d$ is an induced subgraph of the d-dimensional K-cube with uniform radix $j_{d-1} \in \boldsymbol{j}$. K-cubes and K-meshes are identical to traditional cycle-based cubes *resp.* meshes if and only if the maximum (*i.e.*, cube) radix j_{d-1} is either 2 or 3 ([42] Eqn (22)).

The *dimension* of a Hamming graph G is the fewest number of digits that suffice to Hamming label G. Suppose that graph G is Hamming-labeled with majorized radix $\boldsymbol{j}$ on d digits. We say that $\boldsymbol{j}$ is a (minimum) *mesh radix* of G if, for any other (majorized d-digit) radix $\varphi \neq \boldsymbol{j}$ that Hamming labels G: either i) $j_i < \varphi_i$, or ii) $j_i = \varphi_i$ and $\boldsymbol{j}$ is a mesh radix in the $i-1$ lower order digits, as i is decremented from $d-1$ to 0. In other words, if, instead of treating $\boldsymbol{j}$ and φ as vectors, we were to view them as integers radix j_{d-1}, then we would declare $\boldsymbol{j} < \varphi$. Equivalently, a d-digit mesh radix $\boldsymbol{j}$ is *minimum* if, for any other (majorized d-digit) radix $\varphi \neq \boldsymbol{j}$ that Hamming labels G: $j_i \leq \varphi_i$, for *all* $d-1 \geq i \geq 0$ ([31] Cor. 2.4). The largest value of digit j_{d-1} in a mesh radix $\boldsymbol{j}$ is its *cube* radix (hence, j_{d-1} is the cube radix of G). If all the digits of $\boldsymbol{j}$ have the same maximum value $j-1$, then j is the *uniform cube radix* of G. If G is not Hamming then its dimension is infinite.

A subgraph Q of G is *edge-induced* if Q spans G. Recalling that a j-hyperedge, or j-edge, is just a j-vertex clique K_j, a *hyperedge k-separator* of a connected graph G is a subset H of the hyperedges of G (which may include edges, whence $j = 2$) whose removal from G edge-induces $k > 1$ components. Note that we don't remove vertex x in the edge-induced subgraph, even if all of x's neighbors belong to a hyperedge in the inducing separator. We may drop the qualifying k- when the number of components is immaterial.

Our ability to compactly lay out planar electrical circuits derives in large part from a quintessential result in the mathematics of connectivity: the *Lipton-Tarjan Theorem* (10) epitomizes the application of separators (albeit *vertex* separators) to VLSI design software. A graph is *planar* if it can be drawn in the Euclidean plane, with edges represented as uncrossing line segments ([7] p. 22, also p. 10, this paper).

A planar graph G contains at most $2(2n)^{½}$ vertices, deletion of which results in two disconnected parts, neither of which has more than $2n/3$ vertices. ([55] Sec. 3.5) (10)

To conclude our litany of definitions, and recalling page 7's refresher on covers and packings, a *hyperedge matching* is a set of independent hyperedges. A *matched* hyperseparator is therefore a separator comprising disjoint hyperedges. A *(weak) one-hyperfactorization* (WOH) $\boldsymbol{H}$ factorizes a graph into matched hyperseparators $\{H^{j(d-1)}, \ldots, H^{j(0)}\}$. Since our development hinges not on vertex separators, but, rather, on (hyper)edge separators, we may abbreviate the latter, as convenient, by dropping the qualifiers *hyper* and *edge*.

Hamming Graphs: Benefits and Known Properties — LaForge, Korver, and Fadali ([31], [35], [42]) detail how K-cubes rise to the multivariate challenge posed by (9). To summarize: K_j^d has connectivity $f+1 = d(j-1)$, and size $½ \cdot d(j-1) \cdot j^d$, with the former maximum, and the latter minimum, in that they achieve the Harary-Hayes Bound (3). Deleting any $i < d(j-1)$ vertices of K_j^d induces a connected subgraph (quorum) with diameter d or $d+1$. In a ratioed asymptotic sense ([42] Cor. 13.2), the quorum diameter converges to the Moore Bound (8), whenever the number i of vertices deleted guarantees a connected subgraph, and $d \in o(j)$. That is, K-cubes solve (9) for worst-case tolerance f that is superlogarithmic, but sublinear, in n.

Furthermore, K-cubes are preferable to traditional C-cubes (*cf.* definition, p. 20) for the following reasons. i) The *radius* of a C-cube quorum exceeds the *diameter* of a quorum of a comparable K-cube having identical fault tolerance; ii) there is *no* relation such that, as $n \to \infty$, the ratio of the C-cube quorum radius to the Moore Bound does *not* diverge; *i.e.*, this ratio must approach infinity; iii) the *fractional* fault tolerance of K-cubes (*i.e.*, the maximum number of faulty nodes divided by the order of the cube) exceeds that of C-cubes. With respect to each of these criteria, K-cubes are optimal, while C-cubes are suboptimal ([42] Fig. 6).

Graphs which optimize latency, fault tolerance, and throughput enjoy *many short IDJ paths* between vertices. To gauge such graphs with precision, (9) invokes criteria more stringent than those of Section 2.2. To elucidate the bloated eccentricity of Harary-Hayes topologies, it suffices to analyze the case of *no faults*. As Figures 9 and 14 of [44] illustrate, a more thorough treatment would bracket the eccentricity of quorums induced from $G_{\text{H-H}}(n, f)$, as we delete vertices in quantities ranging from zero to f. Herein key insights sprout from *tuning* theorems with this form: "Between two vertices distance k apart in graph G, there are $q(k)$ IDJ paths of length $r(k)$". The crux is to relate k, q, r, and G – a prospect even more daunting than that of formulating distances in graphs with prescribed structure (*cf.* Sub-appendix A.2), but which have not been subjected to vertex deletion. Bose *et al.* [9] and LaForge *et al.* ([42] Thms 5, 12) exemplify tuning theorems in the case of K-cubes and traditional C-cubes. It remains for researchers in the mathematics of connectivity to serve up analogous results for $G_{\text{H-H}}(n, f)$.

a) Are there Hamming graphs other than K-cubes and K-meshes? If so, b) do such graphs necessarily attain multivariate optima for latency, fault tolerance, throughput, and cost – in a scalable fashion, as with K-cubes? *Yes* and *No*.

a) *Yes*. For examples: i) If $j \cdot d > 2$ then deleting any vertex from K_j^d induces a subgraph which remains Hamming labeled (hence is Hamming [31] Thm 4). ii) Any even cycle $G_{\text{H-H}}(2q, 1)$ is Hamming with dimension q and radix 2 ([31] Thm 6). iii) Any n-vertex tree is Hamming with dimension $n-1$ and radix 2 ([31] Thm 7).

b) *No*. Trees, for example, are Hamming. Trees include paths, which maximize eccentricity (*cf.* MESTs, p. 12).

Further, not all graphs are Hamming. For example, if $q > 1$ then any odd cycle $G_{\text{H-H}}(2q+1, 1)$ cannot be Hamming. ([31] Thm 5). More generally:

A triangle-free graph is Hamming only if it is bipartite. (11)

To illuminate (11), we invoke LaForge's characterization of *Hamming ideals* ([31] Cor. 2.4, 2.5):

G is Hamming if and only if G is connected and *monotonically induced* from its Hamming graph ideal. More precisely, suppose G has Hamming dimension d, mesh radix $\boldsymbol{j}$, and cube radix j. Then $K_{\boldsymbol{j}}^d$ (*resp.* K_j^d) constitutes the unique mesh (*resp.* cube) *ideal* of G: that is, the smallest complete Hamming graphs from which we can delete vertices in sequence, such that each of the induced subgraphs is Hamming with respect to the original ideal labeling, with *constant*, unchanging dimension and radix. (12)

From (12) it immediate follows:

The cube radix j of Hamming graph G equals the clique number of G; *i.e.*, the order j of the largest clique in G. (13)

Since any clique larger than an edge contains a triangle, a triangle-free Hamming graph cannot have a clique larger than 2, hence must be binary. To at last establish (11), we appeal to Aurenhammer and Hagauer ([3] Lemma 4):

A binary Hamming graph is bipartite. (14)

As a concrete example, consider 10 vertices labeled as unordered distinct pairs from {0, 1, 2, 3, 4}. Joining two vertices if and only if their labels are disjoint yields the 15-edge, 3-connected *Petersen Graph* G_{P}, whose girth (*i.e.*, shortest cycle) equals five ([8] p. 106). In particular, G_{P} contains no triangle; so if it is Hamming then it must be bipartite. But G_{P} has girth five, so it cannot be bipartite. Therefore

The Petersen graph G_{P} is not Hamming (15)

Furthermore, the Petersen graph perfectly achieves the Moore Bound (9), in the sense described on page 12 (actually, G_{P} is one of at most four perfect Moore graphs of diameter 2). Thus, G_{P} provides an example of a graph which is not Hamming, but which satisfies multivariate challenge (9).

2.4. *Anatomy Synergistic with Physiology: A Hamming Network Is Its Own Distributed Routing Table*

As Section 2.3 points out, some Hamming graphs (*e.g.*, trees, even cycles) are sparse, others (*e.g.*, K-cubes and K-meshes, perhaps with one vertex missing) are dense. Some classes of Hamming graphs (*e.g.*, K_j^d, $j \to \infty$, $d \in o(j)$) are well-suited as scalable network topologies satisfying (9). Others (*e.g.*, K_j^d, $j \in \Theta(1)$, $d \to \infty$) diverge from (8), hence deliver suboptimal latency. Some graphs (*e.g.*, odd cycles) are not Hamming. Beyond G_{P}, moreover, there are graphs which are not Hamming, but which satisfy (9). The latter include those mentioned at the bottom of page 12; *i.e.*, minimum-size graphs, with connectivity and fault tolerance a constant fraction of the order n, and quorum diameter minimized at 2 or 3.

Some Hamming graphs minimize eccentricity, hence tune network or multi-processor *anatomy*. On the other hand, *all* Hamming graphs facilitate latency reduction, as well as throughput enhancement, for reasons of *physiology*.

More precisely, *switch fabric* latency synergistically reciprocates throughput, a local property that may not necessarily extend globally across the quorum (*cf.* [28] Fig. 11). Herein any Hamming-labeled topology can benefit from an optimal, distributed algorithm for *routing*: forward a packet to an adjacent neighbor whose label minimizes the Hamming distance to the destination. As detailed in [31], Algorithm $\boldsymbol{A}_{\text{H-Route}}$ minimizes the execution time of switch fabric step (ii), described at the top of page 9. Embedding $\boldsymbol{A}_{\text{H-Route}}$ effects a local change to each switch fabric, with physiologically global benefits across the quorum. Riding on top of a Hamming topology, moreover, Hamming routing in large part obviates the need for a quorum to map (or remap) its own topology, represent the topology in routing tables, and propagate updates to tables cached in switch fabrics.

To sharpen this point, contrast Hamming routing with conventional routing, wherein nodes cache tables that map how to ship packets. The local storage on each routing node contains information that is, by and large, global. Such schemes suffer *data stale*. Tables become inaccurate with changes in network topology or state. The combination of misdirected packets and table update traffic saps our channel capacity with double-edged penalties, especially in the case of mobile *ad hoc* networks (MANETs). Moreover, why embody a wide-area map in the first place, when all that a switch fabric can really do is schedule the egress channel for each packet? Rather, and in combination with Hamming topologies, Hamming routing puts the *network or multi-processor itself* to work as a distributed routing table. Nodes propagate address updates only when the topology re-tunes itself (*cf.* page 13), and then only to addresses that must be updated.

The preceding sprouts hope that, despite anatomically bloated latency, Harary-Hayes graphs might offer the physiologically redeeming benefit of Hamming routing. Alas, and as Sub-appendix A.5 details, the answer is "Not really". The *only* Harary-Hayes graphs which are Hamming lie at the sparsest and densest extremes of the tunable range of fault tolerance and connectivity. To wit: *cycles* on an even number of vertices, and *cliques*. On the other hand, while traditional C-cubes and C-meshes also exhibit suboptimal eccentricity, they are – and as Sub-appendix A.7 proves – always Hamming, hence always support Hamming routing.

2.5. *Recognizing and Labeling Hamming Graphs: From Theorems to Algorithms*

Complementing Sections 2.3 and 2.4, Aurenhammer and Hagauer [3] articulate additional advantages of Hamming graphs, along with reasons why we should understand them. In particular, and as a potent adjunct to *ad hoc* arguments about whether it is possible to Hamming label an instance (*e.g.*, the Petersen graph), or an instance class (*e.g.*, cycles, trees, Harary-Hayes graphs, C-cubes), we seek to

Give a correct, efficient algorithm that either Hamming labels arbitrary graph G with minimum dimension and radix, if G is Hamming, or declares G not Hamming. (16)

Such an algorithm follows from Theorem 5 of Sub-appendix A.6. In the interest of readability, we present it in English.

Represent the adjacency of (connected) graph G by a two-dimensional array, with entry (i,j) nonempty if and only if $(i,j) \in E(G)$. An entry in row i is doubly linked to the previous and next vertex in the adjacency of i, columnwise doubly linked in a similar fashion. This allows us to check, in constant time, whether (i,j) exists; it also allows us to iterate through the adjacency of a vertex in constant time per neighbor. We can (and will) also remove neighbors from a copy of such a representation, in constant time per edge.

Sub-Algorithm $\boldsymbol{A}_{\text{Enlarge-Edge}}$ — By Corollary 2, if G is Hamming then maximal cliques are edge-disjoint, which we compute as follows. For given vertex x, determine whether neighbors of x, say y_1 and y_2, share an edge. If so, then put y_1 and y_2 in the same *conditional* hyperedge, say $K(x,2)$. Else put y_1 into $K(x,1)$, y_2 into $K(x,2)$. Inspect y_3, an as-yet unex-

amined neighbor of x, to see if it shares an edge with the lowest-indexed vertex in $K(x,1)$; if so, then put y_3 into $K(x,1)$; else if y_3 shares an edge with the lowest-indexed vertex in $K(x,2)$ then put y_3 into $K(x,2)$ … Continuing in this fashion, partition the neighbors of x into conditional hyperedges, at most one per neighbor of x. Invoking Lemma 1, we need check only whether an as-yet unexamined neighbor y shares an edge with but one representative of a conditional clique. At the end of this procedure, verify that each conditional hyperedge is, in fact, a hyperedge and, further, that any two conditional hyperedges are edge-disjoint. If not, output "Not Hamming" and halt. If so, then mark each conditional hyperedge as verified, and remove each edge in each verified hyperedge from the array which represents as-yet unexamined edges. At the end of $\boldsymbol{A}_{\text{Enlarge-Edge}}$ we have examined all pairs of vertices in the adjacency of x, and in time $\Theta|\partial(x)|^2$.

Sub-Algorithm $\boldsymbol{A}_{\text{Rectify-Edges}}$ — Apply $\boldsymbol{A}_{\text{Enlarge-Edge}}$ to identify all hyperedges of G. If G is Hamming with dimension d then $\boldsymbol{A}_{\text{Rectify-Edges}}$, applied to the cube ideal of G, completes in $O(d)$ steps per edge, an amortized cost which does not increase for monotonically induced Hamming subgraphs. Hence, when the input is indeed Hamming, $\boldsymbol{A}_{\text{Rectify-Edges}}$ runs in overall time $O(d|E|)$.

Sub-Algorithm $\boldsymbol{A}_{\text{Hyperfactorize}}$ — Recognize and label connected candidate graphs that are *edge-induced* from Hamming graphs. By Theorem 5, such graphs factorize into hyperseparators. **Algorithm** $\boldsymbol{A}_{\text{Label-Hamming}}$ performs a final check to verify that this labeling is correct, in which case G is trivially edge-induced from a Hamming graph by deleting no vertices.

If G is a lone vertex x then attach the null string to x. Else run j synchronized breadth-first searches (BFS's) from the j vertices of any (unmarked) j-edge, labeling the low-order digit on each vertex with the index of the respective BFS. It is convenient to select j as the radix of the largest hyperseparator not yet marked. Mark the starting j-edge as belonging to hyperseparator 0, corresponding to the 0^{th} digit in our labeling. Figure 3b illustrates.

Advancing to distance i, set the 0^{th} digit in BFS q to q. Vertices x and y from BFS q and r may share an edge, whence mark edge (x,y), as part of separator 0, not to be used again. In this case (and recalling Algorithm $\boldsymbol{A}_{\text{Rectify-Edges}}$) we check if (x,y) is part of a hyperedge with vertices not yet touched by the synchronous BFS, and expand the BFS to proceed from as-yet untouched vertices as well. If x and y share a *vertex* and are not part of the same j-edge ($j > 2$), then G cannot have been monotonically induced from a cube ideal; output "Not Hamming" and halt. If x crosses a separator with more than one edge then G violates the matching condition of Theorem 5: output "Not Hamming" and halt. Subsume both of these conditions by testing whether, at distance i, we try to set the 0^{th} digit of a vertex reached to more than one value. Continuing in this fashion traces a hyperedge separator of j components. Repeating this hyperedge separation, for digits 1, … $d-1$, and on as-yet unmarked (hyper)edges Hamming labels G with the fewest digits d, least radix, and each edge covered by exactly one hyperseparator. Reverse the labels to majorize them. When G is Hamming, we perform d searches, examining each edge $O(d|E|)$ times. Thus:

$$\boldsymbol{A}_{\text{Hyperfactorize}} \text{ correctly hyperfactorizes } G \text{ in time } O(d|E|)$$
$$\text{Else declares } G \text{ not Hamming in time } O(n|E|) \qquad (17)$$

Algorithm $\boldsymbol{A}_{\text{Label-Hamming}}$ — To any candidate labeled by $\boldsymbol{A}_{\text{Hyperfactorize}}$, we can now efficiently apply conditions (a) through (c) of Theorem 5. For convenience we assume an intermediate processing step which augments our representation of the adjacency of G. This augmented data structure allows constant time association of the original label of a vertex with its candidate Hamming label.

Pivoting (say) from high to low order digit in the candidate majorized label $\boldsymbol{x} = (x_{d-1}, \ldots, x_0)$ of vertex x, we test (in constant time) whether, for $y_{d-1} \neq x_{d-1}$, $y = \boldsymbol{y} = (y_{d-1}, \ldots, x_0) \in G$ and, if so, whether edge $(x,y) \in G$. If y is in G, but (x,y) is not in G, then G is not monotonically induced by deleting vertices from a mesh ideal; output "Not Hamming" and halt. (Alternatively, insert the offending edge, flag the repair, and continue). As long as this test passes (or as long as we effect single-edge repairs), repeat this step for every value of the $(d-1)^{\text{st}}$ digit that is not equal to x_{d-1}, and with respect to the mesh radix $\boldsymbol{j} = (j_{d-1}, \ldots, j_0)$ computed by $\boldsymbol{A}_{\text{Hyperfactorize}}$. More generally, iterate through all d digits in the candidate labeling of x, $j_k - 1$ times in the case of x_k. At constant time per test, the aggregate cost of all such tests, run to completion over all nodes when the candidate labeling is in fact Hamming, grows no larger than the number of vertices n times the degree $(j-1)$ of a vertex in the mesh ideal.

$$\text{In time } O(d \cdot \max[\,|E|, n(j-1)],$$
$$\boldsymbol{A}_{\text{Label-Hamming}} \text{ correctly Hamming labels } G;$$
$$\text{Else } G \text{ declares not Hamming, in time } O(n|E|) \qquad (18)$$

Any algorithm is output limited, so, when G is Hamming, the complexity of (16) is $\Omega(dn+|E|)$. Herein $\mathbf{A}_{\text{Label-Hamming}}$ is optimal in the sparse case when $|E| \in \Theta(n)$ and $j \in \Theta(1)$.

When G is Hamming but dense in edges then $|E|/n \in O(j)$. In such a case the running time of $\mathbf{A}_{\text{Label-Hamming}}$ is optimal to within a factor j; this is arguably still efficient, however, since j is at most the d^{th} root of n. Moreover, $\mathbf{A}_{\text{Label-Hamming}}$ is as fast or faster than the most efficient algorithms previously published. If we restrict (16) to *binary* Hamming graphs for example, then the guaranteed execution of Aurenhammer and Hagauer's $O(n \cdot \min[|E|, n \log n])$ algorithm [3] is slower than $\mathbf{A}_{\text{Label-Hamming}}$: $O(n^2 \cdot \log n)$ *vs.* $O(n \cdot \log^2 n)$ in the case of one vertex deleted from a binary cube – and no faster than $\mathbf{A}_{\text{Label-Hamming}}$'s $\Theta(n^2)$ running time, for n-vertex trees. When input G is Hamming, but with unspecified or unbounded radix, the running time of $\mathbf{A}_{\text{Label-Hamming}}$ bests that $O(n^3)$ proffered by Wilkeit [58]. When G is not Hamming, then $\mathbf{A}_{\text{Label-Hamming}}$ runs as fast as the performance guarantee for the algorithm of Aurenhammer and Hagauer, and strictly faster than that of Wilkeit.

3. Recap, Unfinished Business

Multi-processors and networks – especially those destined for spaceflight missions – warrant connectivity *tuned* from sparse to dense. To optimize across a range of priorities for fault tolerance, throughput, latency, and channel cost (including power), the seven applications described in Sections 1.2 through 2.3 bolster the case for compiling and applying a catalog of theorems and algorithms from the mathematics of connectivity. Beyond static papers (such as this one) such a catalog merits *active* manifestation: both as design software, and as realtime embedded software.

Though ripe with solutions for practical application, the mathematics of connectivity hungers for analytic solutions to pressing problems. To this end, this paper answers heretofore open questions: A.2) What formula explicates: i) the distance between given vertices in, and ii) the radius and diameter of an arbitrary Harary-Hayes graph? A.3) What is the maximum diameter of an $(f+1)$-connected graph? A.4) What is the minimum size of a regular graph of prescribed connectivity, such that IID deletion of a constant Bernoulli fraction of vertices induces a connected subgraph? A.5) Which Harary-Hayes graphs are Hamming? A.6) How do hyperseparators prime-factorize Hamming graphs? A.7) What formula explicates the Hamming dimension and radix of traditional C-cubes? $\boldsymbol{A}_{\text{Label-Hamming}}$: how does Theorem 5 translate to a new, efficient algorithm for recognizing and labeling Hamming graphs?

We have also posed a sampling from among the plethora of outstanding challenges: factorization of i) switch fabric schedules and ii) test digraphs (p. 9); iii) complete, exact solution to the Problem of Zarankiewicz (p. 10); closing the factor-of-two gaps in iv) Inequality (7) and v) Theorem 3; vi) *tuning* theorems for Harary-Hayes graphs, as well as others (p. 14); vii) characterization of graphs with constant fractional connectivity, minimum size and eccentricity (p. 12); viii) bridging the algorithmic running time gap for recognizing and labeling Hamming graphs ((16), Section 2.5). Perhaps most importantly, it falls to researchers and practicing engineers to effectively *apply* results for tunable topologies.

Appendix A. Proofs and Derivations

A.1 Lower Bound for the Harary-Hayes Equality (3)

Recalling the discussion on page 5, f-fault tolerance is equivalent to $(f+1)$-connectedness. If some vertex x has degree $\delta < n-1$ then deleting the adjacency of x results in at least two components, one of which comprises x itself. Therefore, the degree of any vertex in an $(f+1)$-connected graph G is at least $f+1$ ([10] Thm 5.1: vertex connectivity ≤ edge connectivity ≤ minimum degree). *I.e.*, the degrees of all n vertices must add up to at least $n(f+1)$. This counts each edge twice, so the total number of edges is at least half the sum. As a finishing touch, the ceiling function $\lceil \cdot \rceil$ takes into account that the size of a G must be an integer: $e_{\min}(n, f) \geq \lceil n(f+1)/2 \rceil$.

A.2 Harary-Hayes Graphs: Distance, Radius, and Diameter

In cycle $G_{\text{H-H}}(n, 1)$ a directed path $P_{(n,1)}(i \ldots j)$ from i to j is (*counter*)*clockwise* if its q^{th} vertex is $[q-i] \bmod n$ (*resp.* $[i-q] \bmod n$). More generally, directed path $P_{(n,f)}(i \ldots j)$ is (*counter*)*clockwise* in $G_{\text{H-H}}(n, f)$ if the vertices of $P_{(n,f)}$ comprise a subset of $P_{(n,1)}(i \ldots j)$ that preserves circular order. That is: $q, r \in P_{(n,f)}$ implies $q, r \in P_{(n,1)}$, with partial pathlengths such that $|P_{(n,1)}(i \ldots q)| < |P_{(n,1)}(i \ldots r)|$ if and only if $|P_{(n,f)}(i \ldots q)| < |P_{(n,f)}(i \ldots r)|$. In cycle $G_{\text{H-H}}(n, 1)$, the distance between the *antipodal* endpoints of any of the edges added at line 9 of algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$ equals the diameter $\lfloor \frac{1}{2} \cdot n \rfloor$ of the cycle, and we will refer to such edges as *antipodal* or, equivalently, *diametric chords*.

Theorem 1. In the Harary-Hayes graph $G_{\text{H-H}}(n, f)$, the edge distance $|i, j|_{\text{H-H}(n,f)}$ between vertices i and j equals

$$\min\Big(\lceil (2/\lceil f+1 \rceil)([j-i] \bmod n) \rceil, \quad \lceil (2/\lceil f+1 \rceil)([i-j] \bmod n) \rceil\Big) \qquad \text{A) odd } f$$

$$\min\Big(\lceil (2/f)([j-i] \bmod n) \rceil, \quad \lceil (2/f)([i-j] \bmod n) \rceil, \quad 1+\lceil (2/f)([j-\underline{i}] \bmod n) \rceil, \quad 1+\lceil (2/f)([\underline{i}-j-(n \bmod 2)(1-\lceil i/n \rceil \lceil j/n \rceil)] \bmod n) \rceil\Big) \qquad \text{B) even } f$$

$$\text{where} \quad \underline{i} = (i + \lceil \tfrac{1}{2} \cdot n \rceil) \bmod n$$

Proof. For (A) note that $[j-i] \bmod n$ is the length of clockwise path $P(i, [(i+1) \bmod n], \ldots j)$, while $[i-j] \bmod n$ is the pathlength of $P(i, [(i-1) \bmod n], \ldots j)$, counterclockwise from i to j. Hence the formula holds at $f = 1$. Otherwise, $f > 1$ and we can iteratively compress the clockwise path by replacing, with a single edge, the $\lfloor f+1 \rfloor/2$ edges from $[i + \frac{1}{2} \cdot (f+1)(k-1) \bmod n]$ to $[i + \frac{1}{2} \cdot (f+1)k \bmod n]$, $1 \leq k \leq \lfloor (2/\lceil f+1 \rceil)([j-i] \bmod n) \rfloor$. If $0 \neq (j-i) \bmod f+1$ then replace with a single edge the $(j-i) \bmod f+1$ edges remaining from the original clockwise path. Algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$ assures that we can perform this compression, thus yielding a shortest path P^+ from i to j, clockwise and of length $\lceil (2/\lceil f+1 \rceil)([j-i] \bmod n) \rceil$. Similarly, we fashion a shortest path P^- from i to j, counterclockwise and with length $\lceil (2/\lceil f+1 \rceil)([i-j] \bmod n) \rceil$.

Without loss of generality, suppose that P^+ is no longer than P^- and that, from i to j, there is a shortest path $P^{\pm}$, strictly shorter than P^+. Path $P^{\pm}$ must change from counterclockwise to clockwise at least once. Hence, there is in $P^{\pm}$ a subpath of vertices $P(x, y, z)$, with edge (x, y) counterclockwise, (y, z) clockwise. By $\boldsymbol{A}_{\text{Construct-G-H-H}}$, vertex z must also be a neighbor of x. Replacing with (x, z) edges (x, y) and (y, z) yields a path from i to j, with length one less than that of $P^{\pm}$. But this contradicts the presumed minimality of $P^{\pm}$. Hence, any minimum length path is either clockwise or counterclockwise. Therefore, at least one of P^+ or P^- achieves least length, among all paths between i and j. Recalling from Table 1 the definition of edge distance, this establishes case (A).

For (B) apply compression, as for case (A), to the subgraph $G_{\text{H-H}}(n, f)$ that discounts antipodal chords. This results in a clockwise path P^+ of length $\lceil (2/f)([j-i] \bmod n) \rceil$, along with a counterclockwise path P^- of length $\lceil (2/f)([i-j] \bmod n) \rceil$. Hence, any path from i to j that does not traverse antipodal chords is at least as long as the shorter of P^+ and P^-. This gives the first two expressions minimized in (B). It remains to consider antipodal *paths*, *i.e.*, paths that contain antipodal chords.

To preview: we first observe that any shortest antipodal path can be replaced by an antipodal path which is at least as short, and which traverses all antipodal chords in succession. The upshot is that, with but one exception, a shortest antipodal path from i to j contains only one antipodal chord, which may just as well contain an endpoint of the path. From the antipode of i to j, the remainder of such a shortest path is clockwise or counterclockwise. Bookkeeping similar to that for (A) explicates expressions (B) for the pathlength.

Direct from i to j any shortest antipodal path P, such that we traverse at least one antipodal chord (x, y) preceded and succeeded by non-antipodal subpaths $P(w \ldots x)$ and $P(y \ldots z)$. By a contraposition argument similar to that employed for case (A), $P(w \ldots x)$ and $P(y \ldots z)$ must either both be clockwise, or both counterclockwise (else the length of P is not minimum). Without loss of generality suppose that $P(w \ldots x)$ and $P(y \ldots z)$ are both clockwise. Replace $P(w \ldots x) \cup \{(x, y)\} \cup P(y \ldots z)$ with $P(w \ldots x) \cup P(\underline{y} \ldots \underline{z}) \cup \{(\underline{z}, z)\}$, where q and $\underline{q}$ are antipodal. Algorithm $\boldsymbol{A}_{\text{Construct-G-H-H}}$ assures that we can perform this replacement, thus producing an antipodal path from i to j, with length equal to that of P. Continuing in this fashion results in a shortest antipodal path that traverses all antipodal chords in succession, and at one end of the path. By $\boldsymbol{A}_{\text{Construct-G-H-H}}$, the only instance with more than one antipodal chord occurs when both n and $f+1$ are odd, and vertex 0 is antipodal with $i \in \{\lfloor \tfrac{1}{2}\cdot n \rfloor, \lceil \tfrac{1}{2}\cdot n \rceil\}$. For this exception, replace the two antipodal chords by a single edge $(\lfloor \tfrac{1}{2}\cdot n \rfloor, \lceil \tfrac{1}{2}\cdot n \rceil)$. But this shortens by one the pathlength, contradicting the minimality of the path so modified. Therefore, any shortest antipodal path from i to j contains exactly one antipodal chord, which we may, without loss of generality, take as the first edge traversed.

For Case (B) therefore, a shortest antipodal path from i to j consists of an antipodal chord followed by a shortest clockwise or counterclockwise path from $\underline{i}$ to j. However, if both n and $f+1$ are odd and one of i or j is zero, then, to obtain a shortest path, we need to pick $\underline{i} = \lfloor \tfrac{1}{2}\cdot n \rfloor$ as our antipode; hence the appearance of $(n \bmod 2)(1-\lceil i/n \rceil \lceil j/n \rceil)$ in the last expression minimized by (B). As a bookkeeping check, and recalling property (ii) of the *metric space* entry in Table 1, verify that the formulae for (A) and (B) are *symmetric*. That is, the *set* of values of the pertinent expressions to be minimized does not change if we interchange i and j. Fourteen expressions govern the possible cases: A) f odd; B.1) f and n both even; B.2) f even, n odd, and one of i or j equal to zero; B.3) f even, n odd, and neither i nor j equal to 0. ❐

For any vertex x of $G_{\text{H-H}}(n, f)$, Theorem 1 enables us to compute $\text{Ecc}(x)$. For cases (A) and (B.1), we may, without loss of generality, take $x = 0$. Applying the expressions for (A), $\min(j \bmod n, -j \bmod n)$ increases as j varies from 0 to $\lfloor \tfrac{1}{2}\cdot n \rfloor$, maintains the same value for $j = \lceil \tfrac{1}{2}\cdot n \rceil$, whence decreases as j approaches $n-1$. Therefore, the maximum eccentricity of *any* vertex is given by substituting into (A) the values $i = 0, j \in \{\lfloor \tfrac{1}{2}\cdot n \rfloor, \lceil \tfrac{1}{2}\cdot n \rceil\}$.

For case (B.1) n is even, so it suffices to consider just the first and fourth expressions with $0 \le j \le \tfrac{1}{2}\cdot n$. *I.e.*, we can obtain the maximum eccentricity of vertex 0 by reducing $G_{\text{H-H}}(n, f)$ to $G_{\text{H-H}}(\tfrac{1}{2}\cdot[n+f], f-1)$: i) induce a subgraph by deleting the $\tfrac{1}{2}\cdot n - 1$ vertices whose indices range from $\tfrac{1}{2}\cdot n + 1$ to $n-1$. This excises all antipodal chords, save for $(0, \tfrac{1}{2}\cdot n)$. ii) Replace $(0, \tfrac{1}{2}\cdot n)$ with $\tfrac{1}{2}\cdot f - 1$ vertices, indexed as $\tfrac{1}{2}\cdot n + 1, \ldots, \tfrac{1}{2}\cdot(n+f) - 1$. iii) Ensure the existence of edge (i, j), which may have to be introduced, whenever $[i - j] \bmod n$ is less than or equal to $\tfrac{1}{2}\cdot f$. By the preceding paragraph, the maximum eccentricity is attained at $j \in \{\lfloor \tfrac{1}{4}\cdot(n+f) \rfloor, \lceil \tfrac{1}{4}\cdot(n+f) \rceil\}$.

Solve the remaining cases with similar reductions. For (B.2), $j = \lfloor \tfrac{1}{2}\cdot\lfloor \tfrac{1}{2}\cdot n \rfloor + \tfrac{1}{4}\cdot f = \tfrac{1}{4}\cdot(n+f-1) \rfloor$ maximizes the eccentricity of $G_{\text{H-H}}(\tfrac{1}{2}\cdot[n+f-1], f-1)$. $G_{\text{H-H}}(\tfrac{1}{2}\cdot[n+f+1], f-1)$ achieves a maximum at $j = \lfloor \tfrac{1}{2}\cdot\lceil \tfrac{1}{2}\cdot n \rceil + \tfrac{1}{4}\cdot f = \tfrac{1}{4}\cdot(n+f+1) \rfloor$. which answers (B.3). To wrap up:

Corollary 1. $G_{\text{H-H}}(n, f)$ has eccentricity given by

$\text{Rad} = \text{Diam} =$	$\lceil (2/\lfloor f+1 \rfloor)\lfloor \tfrac{1}{2}\cdot n \rfloor \rceil$	A)	odd f
$\text{Rad} =$	$\lceil (2/f)\lfloor \tfrac{1}{2}\cdot\lfloor \tfrac{1}{2}\cdot n \rfloor + \tfrac{1}{4}\cdot f \rfloor \rceil$	B.1, B.2)	even f
$\text{Diam} =$	$\lceil (2/f)\lfloor \tfrac{1}{2}\cdot\lceil \tfrac{1}{2}\cdot n \rceil + \tfrac{1}{4}\cdot f \rfloor \rceil$	B.2, B.3)	even f

A.3 The Maximum Diameter of an $(f+1)$-Connected Graph

By reasoning similar to that of Sub-appendix A.1, if G is an $(f+1)$-connected graph then there must be at least $f+1$ vertices at any edge distance $i < k$ from vertex x, between x and any other vertex y, with $|x, y| = k$. Otherwise, deleting f or fewer vertices edge distance i from x would separate x and y into distinct components, contradicting the presumed $(f+1)$-connectedness of G. By maximizing k, we conclude:

Theorem 2. If G is $(f+1)$-connected then, achievably,

$$\text{Diam}(G) \le \lfloor (n+f-1)/(f+1) \rfloor$$

To see that the upper bound above is best possible, verify that the following construction, designated by $G_{\text{LMF}}(n, f)$ and illustrated in Figure 3a, achieves equality for any $n \ge f+3$. Interpose $\lfloor (n-2)/(f+1) \rfloor$ clique-blocks between x and y, with every vertex in block i edge distance i from x. Put $(f+1) + [(n-2) \bmod (f+1)]$ vertices into block $i = \lfloor (n-2)/(f+1) \rfloor$. Populate each of the remaining blocks with $f+1$ vertices. Number the vertices of any blocks with successive integers, beginning at zero. In any block i, connect vertex w to every other $z \in \{0, \ldots, f\}$, and to any vertex with the same index z in a neighboring block $i \pm 1$. Join x to vertices 0 through f in block 1 (which may also be closest to y). From y draw an edge to each vertex z of block $\lfloor (n-2)/(f+1) \rfloor$ where $z \in \{0, \ldots, f\}$. ❐

For $G_{\text{LMF}}(n, f)$ verify: a) there are $f+1$ independent paths between x and y (each edge has at most one endpoint in any block); b) f independent paths between any two vertices within a block, and at least one passing through a neighboring block; c) $f+1$ independent paths between any two vertices of different blocks. Hence the $G_{\text{LMF}}(n, f)$ is $(f+1)$-connected. Simplifying expressions bracket the edgecount:

$$\begin{aligned} &\tfrac{1}{2}\cdot[(f+2)(n-2) + (f+1)] \\ &\quad \le |E(G_{\text{LMF}}(n, f))| \\ &\quad\quad = \lceil \tfrac{1}{2}\cdot(f+1)(2 + [f+2]\lfloor (n-2)/(f+1) \rfloor \\ &\quad\quad\quad\quad + 2[n-2 \bmod f+1] \rceil) \\ &\quad\quad\quad \le \tfrac{1}{2}\cdot[(f+2)(n-2)] + [f+1]^2 \end{aligned} \qquad (19)$$

A.4 Probabilistically Tuned Fault Diagnosis and Tolerance

In this paper we use *almost surely* to mean *with high probability* $1-o(1)$. Stronger than *convergence in distribution*, but weaker than *convergence in measure* (which some authors use synonymously with *almost sure*), our use of *convergence in probability* aligns with results for very large scale integrated systems (VLSI) and graph theory. Traditionally, the latter seeks the fewest IID edges such that, with high probability, a random graph of order n exhibits some particular property; *e.g.*, connectedness, Hamiltonicity, or the emergence of a clique of prescribed order [7]. By contrast, we ask for *some* graph having fewest edges, such that *stochastic fault injection* – modeled by the deletion of an IID fraction of vertices – induces a subgraph which almost surely possesses beneficial properties; *e.g.*, connectedness or low eccentricity.

Perhaps the starkest distinction between random graphs and stochastic subgraphs arises with Blough's counterexample $G_B(n, p, h)$ to Scheinerman's claim that $\Omega(n \log n)$ tests are necessary to almost surely diagnose faults distributed with Bernoulli probability p [6], [41], [54]. Connect h vertices V_h in $G_B(n, p, h)$ to the remaining $n-h$ nodes $V \setminus V_h$. An induced subgraph forms a single connected component if and only if at least one vertex in V_h is healthy. The probability $1-p^h$ of this event approaches 1 if and only if $h(n) = \lceil \log_{1/p} \omega'(n) \rceil$. Diagnose as healthy the nodes of this component, whence neighboring vertices necessarily declare each other to be healthy. $G_B(n, p, h)$ is almost surely diagnosable with respect to PMC conditions: i) healthy nodes tell the truth; ii) faulty nodes may lie; iii) a minority of nodes are faulty; *i.e.*, $p < ½$. The particular function $\omega(n)$ *tunes* how fast the probability of correct diagnosis approaches one. *E.g.*, if $\omega'(n) = n$ then the probability of diagnosing and forming a quorum is at least $1-n^{-1}$. Since $\lceil \log_{1/p} \omega'(n) \rceil \in \omega(n)$, G_B has size $\Theta(n \cdot \omega[n])$. To see that this is the best for which we could hope, note that having some vertex degree bounded by a constant δ_{const} implies that the probability of no isolated vertex is at most $(1-p)(1-p^{\delta const}) < 1-p < 1$, whence almost sure diagnosis or tolerance is unattainable. Thus, $\delta(x) \in \omega(n)$ for any vertex x, and $\Theta(n \cdot \omega[n])$ tests or edges is the least cost of almost sure diagnosability and tolerance.

Now suppose we want to minimize the test or edge count, with maximum vertex degree at most δ^+; this constraint is likely when the number of channels supported by any node is limited; *e.g.*, in a sensor network or multi-processor. By Section 1.2, minimum edge count fault tolerance implies topologies that are regular, or nearly so. For given edge count, that is, we maximize the worst-case fault tolerance (equivalently, we maximize the connectivity) only if the maximum degree minus the minimum degree is at most one. For combined almost sure and worst-case fault diagnosability and tolerance, and neglecting the "stray edge" described on page 5, this calls for graphs which are δ-regular.

Consider an arbitrary δ-connected, δ-regular graph G (hence of minimum edge count), and let I be the random variable counting the number of isolated healthy vertices in a subgraph Q, stochastically induced from G with Bernoulli parameter p. Applying Erdös and Rényi's *method of moments* ([7] p. 232), let us establish a $\log_{1/p} [n/\omega(n)] \in [\log_{1/p} n] - \omega(n)$ lower bound on δ. Write $\hat{\mathbf{E}}$ and VAR for the expectation *resp.* variance operators, with shorthand $\mu = \hat{\mathbf{E}}(I)$ and $\sigma^2 = \text{VAR}(I)$. By the linearity of expectation:

$$\mu = (1-p) n p^{\delta} = (1-p)\omega(n) \to \infty \qquad (20)$$

As the order n of G increases, that is, the expected number of isolated vertices in Q increases faster than any constant, as long as we throttle the degree and worst-case connectivity δ at $\log_{1/p} [n/\omega(n)]$. Let us sharpen (20) by proving that, *almost surely*, Q has many isolated vertices. By Chebyshev's inequality ([15] p. 185), it suffices to show that $\sigma^2 \in o(\mu^2)$. We obtain sufficient bounds on σ^2 by bracketing the covariance $\hat{\mathbf{E}}(I \cdot [I-1])$ between ordered pairs of isolated vertices in Q. Pick an isolated vertex x. By definition, $\partial(x) \notin Q$, so, for any other isolated vertex y of Q, $|x, y| \geq 2$. Also:

$$\delta \leq |\partial(x, 2)| = \Delta \leq \delta(\delta-1) \qquad (21)$$

Where the lefthand side follows by the δ-connectedness of G, the right by δ-regularity. Maximizing Δ in (21) maximizes

$$\hat{\mathbf{E}}(I \cdot [I-1]) = (1-p)^2 n [(n-\delta-1-\Delta) p^{2\delta} + \Delta p^{\delta}] \qquad (22)$$

Whence

$$\begin{aligned} \sigma^2 &= \hat{\mathbf{E}}(I^2) - \mu^2 \\ &\leq \mu [1 + (1-p)(\delta^2 [1-p^{\delta}] - \delta - p^{\delta})] \\ &\in o(\mu^2) \end{aligned} \qquad (23)$$

For quorum Q to be almost-surely connected, therefore, the connectedness and regularity of G must exceed $[\log_{1/p} n] - \omega(n)$. Constructively match this lower bound with an h-redundant *locally-spared* Hamiltonian cycle $G_{H\text{-}L}(n, p, h)$: distribute h out of n vertices into $\lfloor n/h \rfloor$ blocks; if h does not divide n, then put one extra vertex in each of blocks 0 through $[n \bmod h] - 1$. Connect vertices in block i to those in block $(i+1) \bmod \lfloor n/h \rfloor$. LaForge's formula spells out the minimum $h = \lceil \log_{1/p} n \rceil$ such that a stochastically induced quorum of $G_{H\text{-}L}(n, p, h)$ almost surely contains one vertex in each block ([37] Thm 2; [44] Fig. 12a). Since each vertex shares at most $1 + \lceil \log_{1/p} n \rceil$ edges with vertices of the succeeding block, $G_{H\text{-}L}(n, p, h)$ delivers almost sure connectivity, with each degree at most $2(1 + \lceil \log_{1/p} n \rceil)$, *i.e.*, about twice the lower bound on degree $\lceil [\log_{1/p} n] - \omega(n) \rceil$ derived above. We record our development as Theorem 3, a fresh contribution to the mathematics of connectivity. ❒

Theorem 3. Suppose G is a δ-connected, δ-regular graph of order n. Induce subgraph Q by deleting each vertex of G, with constant Bernoulli probability p. Q is almost surely connected only if $\delta \geq \log_{1/p} [n/\omega(n)]$. To within a factor of two, this is the least possible value of δ.

It remains to close the factor-of-two gap between the lower bound of Theorem 3 and that given by $G_{H\text{-}L}(n, p, h)$. We conjecture that, in fact, a $2\lceil [\log_{1/p} n] - \omega(n) \rceil$ upper bound is tightest possible. If so, then the class $G_{H\text{-}L}(n, p, h)$ exemplifies almost-surely-*np*-connected regular graphs of minimum size, akin to Harary-Hayes graphs $G_{H\text{-}H}(n, f)$, minimum size for $(f+1)$-connectivity in the worst case. Whether or not our conjecture is true, we have proved that the minimum per vertex cost is $\Theta(\log n)$ edges, for almost sure tolerance to a constant proportion p of Bernoulli faults, and as claimed in (c) on page 9. Other results in (a) – (d) on page 9 can be ferreted from the literature. We conclude this sub-appendix with remarks to help researchers probe further.

With either (c) on page 9, or irregular test graphs, such as $G_B(n, p, h)$ on the previous page, Blough's algorithms rely heavily on *majority voting* [6]. By contrast, we tend to recommend *connected-component* algorithms, in part because they often reduce diagnosis and configuration to the same problem. Connected-component configuration also correctly diagnoses faulted instances of $G_{H\text{-}L}(n, p, h)$, as well as $G_B(n, p, h)$ on the previous page. With case (d) on page 9, for example, LaForge *et al*. [39] explicate a per node cost of $\Theta(1)$ tests. However, and as Figure 7 of [44] illustrates, the same connected-component algorithm used for almost sure diagnosis also realizes a quorum, as long as the quorum is only required to contain *almost* every healthy node. This should not be too surprising, considering that much of [39] was inspired by Leighton and Leiserson's landmark work [45] on fault tolerant configuration of two-dimensional arrays – using connected components, of course.

A.5 Harary-Hayes Graphs: What's Hamming, What's Not

Lemma 1. Two distinct triangles of a Gray-codable graph G either belong to the same hyperedge, or are edge-disjoint.

Proof. Two triangles share at most one edge (x, y), else they are not distinct. Denote by w and z vertices that distinguish triangles sharing edge (x, y). Proceeding by contraposition, suppose that x, y, w, and z do not all belong to the same hyperedge, whence $(w, z) \notin E(G)$. Labels x, y, and w must be Hamming distance 1 from one another; *i.e.*, differing only in one digit, say i. Similarly, labels x, y, and z must differ only in one digit, say j. Digit positions i and j must be distinct, else the four vertices belong to the same hyperedge. Without loss of generality, let $i_x = 0$, $i_y = 1$, $i_w = 2$. but this implies contradictory simultaneous equalities: $i_z = 0$ and $i_z = 1$. ❐

Corollary 2. Any two hyperedges of a Gray-codable graph are edge-disjoint.

Theorem 4. A Harary-Hayes graph is Hamming if and only if it is an even cycle $G_{H\text{-}H}(2q, 1)$, or a clique $G_{H\text{-}H}(n, n-2)$.

If Sufficiency. By Theorem 6 of [31], $G_{H\text{-}H}(2q, 1) = C_{2q}$ is Hamming with dimension q and radix 2. $G_{H\text{-}H}(n, n-2) = K_n$ is Hamming with dimension 1 and radix n.

Only If Necessity. By Theorem 5 of [31], $G_{H\text{-}H}(2q+1 > 3, 1) = C_{2q+1}$ is never Hamming. If $1 < f < n-2$ then the clique number of a Harary-Hayes graph is 3; whence $G_{H\text{-}H}(n, f)$ violates Lemma 1 for $f > 2$. $G_{H\text{-}H}(n, 2)$ contains an odd cycle when n is odd or n and $\frac{1}{2}n$ are both even, contradicting (13) and (14). Figure 3b illustrates the remaining case. Starting with edge $(0, 1)$ of $G_{H\text{-}H}(2[2q+1], 2)$, the breadth-first search of Algorithm $\boldsymbol{A}_{\text{Label-Hamming}}$ pegs two edges in the dimension separator emanating from vertices $q+1$ and $(-q \bmod 2[2q+1])$. But this contradicts the property of Theorem 5 that the edges of such a separator be matched; *i.e.,* vertex-disjoint. Although it is bipartite, that is, $G_{H\text{-}H}(2[2q+1], 2)$ is not Hamming. ❐

A.6 How Hamming Graphs Factorize into Hyperseparators

Theorem 5 of this sub-appendix augments (12) by pinpointing *computable* properties of Hamming graphs. These properties serve as the foundation of **Algorithm** $\boldsymbol{A}_{\text{Label-Hamming}}$, which not only recognizes and labels Hamming graphs, but which enables us to correctly and efficiently construct Hamming graphs from edge-induced subgraphs of monotonically induced subgraphs of mesh ideals.

Theorem 5. A connected graph G is Hamming with dimension d and radix $\boldsymbol{j} = (j_{d-1}, \ldots, j_0)$, if and only if G prime-factorizes into d matched hyperseparators $\boldsymbol{H} = \{H_{d-1}, \ldots, H_0\}$ such that, a) for vertices x and y in G, b) (x, y) is an edge of G whenever c) (x, y) is an edge in $K_{\boldsymbol{j}}^{\,d}$.

Remark 5.1. Mindful that $\boldsymbol{j}$ is majorized (*cf.* definitions, p. 13), H_k edge-induces j_k components. A vertex of $G_k^{\,i}$ may belong to at most one hyperedge in H_k. The hyperedges of H_k need not all have the same radix, but at least one hyperedge of H_k is a j_k-edge; *i.e.*, a clique $K_{j(k)}$ on j_k vertices.

Remark 5.2. No two matched hyperseparators of $\boldsymbol{H}$, say H_k and H_q, share an edge (*cf.* definitions, p. 14). However, *components* $G_k^{\,i}$ and $G_r^{\,p}$, edge-induced by distinct matched hyperseparators H_k and H_r, may share vertices or edges.

Remark 5.3. Conditions (a) through (c) amount to (12). We therefore need only show that Hamming graphs prime-factorize into matched hyperseparators.

Only If Necessity. Suppose that G is Hamming with dimension d and mesh radix $\boldsymbol{j}$. By (12), G is monotonically induced from its mesh ideal $K_{\boldsymbol{j}}^{\,d}$. Recalling definitions on p. 13, $K_{\boldsymbol{j}}^{\,d}$ comprises a complete (hence prime) factorization into hyperseparators $\boldsymbol{H} = \{H_{d-1}, \ldots, H_0\}$, where H_k is a matching of j_k-edges, and $|H_k| = \prod_{0 \le i \le k-1} j_i$. Monotonically inducing G from $K_{\boldsymbol{j}}^{\,d}$ induces a prime subfactorization which covers G: d matchings, with at least one j_k-edge in the k^{th} such matching, else $K_{\boldsymbol{j}}^{\,d}$ is not the mesh ideal of G. ❐

A.7 C-cubes and C-meshes are Binary Hamming

Construct a *labeled d-dimensional j-ary C-cube* $C_j^{\,d}$ as follows. For $j = 2$ we define $C_2^{\,d}$ to be a d-dimensional binary K-cube $K_2^{\,d}$. For $j > 2$: $C_j^{\,0}$ is a single unlabeled vertex. $C_j^{\,1}$ is a cycle on j vertices, numbered circularly from 0 to $j-1$; two vertices are joined by an edge if and only if the modulo j difference in their labels equals ±1. In general, to construct $C_j^{\,d}$ we i) make j copies of $C_j^{\,d-1}$; ii) prepend i to the label of each vertex of the i^{th} copy of $C_j^{\,d-1}$; iii) connect with an edge vertices u and v (from different copies of $C_j^{\,d-1}$) if and only if the modulo j difference in the high order digits of the labels on u and v equals ±1, and the low order $d-1$ digits are identical. Alternatively, we can label $n_K = j^d$ vertices with integers 0 through $j-1$ along d axes, each corresponding to a digit. Hence $C_j^{\,d}$ is independent of the order in which dimensions are populated. Label the vertices of a d-dimensional $\boldsymbol{j}$-ary C-*mesh* $C_{\boldsymbol{j}}^{\,d}$ with 0 through $j_i - 1$ along the i^{th} axis, where i ranges from 0 to $d-1$ inclusive. Connect two vertices with an edge if and only if the modulo j_i difference between the i^{th} digits of the respective labels equals ±1, and all other digits are identical. Figure 2 illustrates 5-ary C-cubes and C-meshes in 1 and 2 dimensions.

Theorem 6. C-cubes and C-meshes are binary Hamming, with dimension $d(j-1)$, for C_j^d, and $\sum_{0 \le i \le d-1} (j_i - 1)$ for $C_{\mathbf{j}}^d$.

Proof. Applying Theorem 5, let $H_{i,q}$ be the edges separating C-mesh vertices labeled $q-1$ and q in digit i, $1 \le q < j_i$, all other digits identical. On each side of $H_{i,q}$, Hamming label digit $q + \sum_{0 \le r \le i-1} (j_r - 1)$ with complementary bit values. ❐

REFERENCES

[1] L. Alkalai and D. Geer, "Space-Qualified 3D Packaging Approach for Deep Space Missions: New Millennium Program, *Deep Space 1* Micro-Electronics Systems Technologies". Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 7-Oct-1996. Cited p. 1.

[2] L. Alkalai and A. T. Tai, "Long-life Deep-Space Applications". *Computer*. Aug-1998, pp. 37 – 38. Cited p. 4.

[3] F. Aurenhammer and J. Hagauer, "Recognizing Binary Hamming Graphs in $O(n^2 \log n)$ Time". *Math. Systems Theory*. **28** (5), 1995, pp. 387 – 395. Cited pp. 15, 16.

[4] A. Avizienis, "The Hundred Year Spacecraft". *Proceedings, First NASA/DOD Workshop on Evolvable Hardware*. Pasadena, CA: Jet Propulsion Laboratory, Jul-1999, pp. 233 – 239. Cited p. 1.

[5] A. Avizienis, "Toward Systematic Design of Fault-tolerant Systems". *Computer*. Apr-1997, pp. 51 – 58. Cited p. 1.

[6] D. M. Blough, *Fault Detection and Diagnosis in Multiprocessor Systems*. Ph.D. dissertation, Baltimore: Johns Hopkins University, 1988. Cited pp. 9, 20.

[7] B. Bollobás, *Modern Graph Theory*. New York: Springer-Verlag, 1998. Cited pp. 2, 3, 10, 12, 14, 19, 19.

[8] B. Bollobás, *Extremal Graph Theory*. London: Academic Press, 1978. Cited pp. 7, 8, 12.

[9] B. Bose, B. Broeg, Y. Kwon, and Y. Ashir, "Lee Distance and Topological Properties of *k*-ary *n*-cubes". *IEEE Transactions on Computers*. **44** (8), Aug-1995, pp. 1021 – 1030. Cited p. 14.

[10] G. Chartrand and L. Lesniak, *Graphs and Digraphs*. Belmont, CA: Wadsworth, Inc, 1986. Cited pp. 2, 5, 12, 17.

[11] V. Chvatal, "A Greedy Heuristic for the Set Covering Problem". *Mathematics of Operations Research*. **4** (3), Aug-1979, pp. 233 – 235. Cited pp. 7, 8.

[12] P. Christie and D. Stroobandt, "The Interpretation and Application of Rent's Rule". *IEEE Transactions on Very Large Scale Integrated (VLSI) Systems*. **8** (6), Dec-2000, pp. 639 – 648. Cited p. 10.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. Cambridge, MA: MIT Press. Tenth printing, 1993. Cited pp. 8, 12.

[14] M. de Berg, M. van Krevald, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*. 2nd edition. Berlin: Springer-Verlag. 2000. Cited pp. 10, 12.

[15] M. H. DeGroot, *Probability and Statistics*. Reading, MA: Addison-Wesley. 1975. Cited p. 19.

[16] DoD JTRS Joint Program Office, "*Joint Tactical Radio System (JTRS) Wideband Networking Waveform (WNW) Functional Description Document (FDD)*". V 2.21 29-Nov-2001. Cited p. 8.

[17] D. Edenfeld, A. B. Kahng, M. Rogers, and Y. Zorian, "2003 Technology Roadmap for Semiconductors". *Computer*. Jan-2004, pp. 47 – 56. Cited p. 1.

[18] I. Elhaney and V. Tabatabaee. "Benchmarking Next-Generation Switch Fabrics". *Computer*. Oct-2003, pp. 109 – 110. Cited p. 9.

[19] G. D. Forney, Jr., "The Viterbi Algorithm". *Proc. of the IEEE*. **61** (3), Mar-1973, pp. 268 – 278. Cited p. 9.

[20] Z. Galil, "Efficient Algorithms for Finding Maximum Matching in Graphs". *ACM Computing Surveys*. **18** (1), Mar-1986, pp. 295 – 305. Cited p. 7.

[21] M. R. Garey and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-Completeness*. New York: W. H. Freeman and Company, 1979. Cited p. 7.

[22] P. Giaccone, D. Shah, and B. Prabhakar, "An Implementable Parallel Scheduler for Input-Queued Switches". *IEEE Micro*. **22** (1), Jan/Feb-2002, pp. 19 – 25. Online at http://ieeexplore.ieee.org/Xplore/. Cited p. 9.

[23] S. Gupta, *Performance Evaluation of Ad Hoc Routing Protocols Using* ns-2 *Simulations*. Department of Computer Science, University of Tennessee. Undated report. Online at http://www.cs.utk.edu/~gupta/Adhoc.doc. Cited p. 4.

[24] F. Harary, "The Maximum Connectivity of a Graph". *Proceedings, National Academy of Science*. **48**, 1962, pp. 1142 – 1146. Cited pp. 4, 7.

[25] J. P. Hayes, "A Graph Model for Fault Tolerant Computing Systems". *IEEE Transactions on Computers*. **C-25** (9), Sep-1976, pp. 875 – 884. Cited pp. 4, 7.

[26] L. E. Jackson and G. N. Rouskas, "Deterministic Preemptive Scheduling of Real-Time Tasks". *Computer*. May-2002, pp. 73 – 79. Cited p. 9.

[27] D. S. Katz and R. R. Some, "NASA Advances Robotic Exploration". *Computer*. Jan-2003, pp. 52 – 61. Cited p. 1.

[28] M. Kobrinsky, B. Block, J-F. Zheng, B. Barnett, E. Mohammed, M. Reshotko, F. Robertson, S. List, I. Young, and K. Cadien, "On-chip Optical Interconnects". *Intel Tech. Jnl.* **8** (2), May-2004, pp. 128 – 142. Online at http://developer.intel.com/technology/itj/2004/ volume08issue02. Cited pp. 8, 10, 15.

[29] S. Y. Kuo and W. K. Fuchs, "Efficient Spare Allocation for Reconfigurable Arrays". *IEEE Design and Test*. February, 1987, pp. 24 – 31. Cited p. 10.

[30] C.-L. Kwan and S. Toida, "Optimal Fault-tolerant Realizations of Some Classes of Hierarchical Tree Systems". *Proceedings, 11th Annual Symposium on Fault-Tolerant Computing*. 26-Jun-1981, pp. 176 – 178. Cited p. 3.

[31] L. E. LaForge, "Hamming Graphs". *Proc., 2004 IPSI: Internet, Processing, Systems for E-education/E-business, and Interdisciplinaries*. 9-Oct-2004. Cited pp. 7, 13, 14.

[32] L. E. LaForge, *Clique-Factorized Optimum MANET Throughput Enabled by Directed, Power-Controlled Antennas*. Technical report, U.S. DoD contract FA8750-04-C-0020. Reno, NV: The Right Stuff of Tahoe, Incorporated. 14-Apr-2004 revised 7-Aug-2004. Cited pp. 8, 12.

[33] L. E. LaForge, "What Designers of Microelectronic Systems Should Know About Arrays Spared by Rows and

Columns". *IEEE Transactions on Reliability*. Special Issue on Very Large Scale Integrated Systems. **39** (3), Sep-2000, pp. 251 – 272. Expanded version online at http://faculty.erau.edu/laforgel/. Cited p. 10.

[34] L. E. LaForge, "Self-healing Avionics for Starships". *Proceedings, 2000 IEEE Aerospace Conference*. 18-Mar-2000. Cited p. 1.

[35] L. E. LaForge, *Fault Tolerant Physical Interconnection of X2000 Computational Avionics*. Pasadena, CA: Jet Propulsion Laboratory, document number JPL D-16485. 28-Aug-1998, revised 18-Oct-1999. Online at http://faculty.erau.edu/laforgel/. Cited pp. 4, 14.

[36] L. E. LaForge, "Configuration of Locally Spared Arrays in the Presence of Multiple Fault Types". *IEEE Transactions on Computers*. **48** (4), Apr-1999, pp. 398 – 416. Online at http://faculty.erau.edu/laforgel/. Cited p. 4.

[37] L. E. LaForge, "What Designers of Wafer Scale Systems Should Know About Local Sparing". *Proc., 1994 IEEE International Conference on Wafer Scale Integration*. R. M. Lea and S. K. Tewksbury, eds. Los Alamitos: IEEE Computer Society Press, 1994, pp. 106 – 131. Cited pp. 9, 10, 19.

[38] L. E. LaForge, *Fault Tolerant Arrays*. Ph.D. dissertation. Montreal: McGill University, 1991. Cited p. 9.

[39] L. E. LaForge, K. Huang, and V. K. Agarwal, "Almost Sure Diagnosis of Almost Every Good Element". *IEEE Transactions on Computers*. **43** (3), Mar-1994, pp. 295 – 305. Online at http://faculty.erau.edu/laforgel/. Cited pp. 3, 7, 9, 20.

[40] L. E. LaForge and K. F. Korver, "Graph-theoretic Fault Tolerance for Spacecraft Bus Avionics". *Proc., 2000 IEEE Aerospace Conference*. 18-Mar-2000. Cited p. 7.

[41] L. E. LaForge and K. F. Korver, "Mutual Test and Diagnosis: Architectures and Algorithms for Spacecraft Avionics". *Proceedings, 2000 IEEE Aerospace Conference*. 18-Mar-2000. Cited pp. 7, 9.

[42] L. E. LaForge, K. F. Korver, and M. S. Fadali, "What Designers of Bus Structures and Networks Should Know About Hypercubes". *IEEE Transactions on Computers*. **52** (4), Apr-2003, pp. 525 – 544. Online at http://faculty.erau.edu/laforgel/. Cited pp. 5, 12, 13, 14.

[43] L. E. LaForge, J. R. Moreland, R. G. Bryan, and M. S. Fadali, "Vertical Cavity Surface Emitting Lasers for Spaceflight Multi-Processors". *Proceedings, 2006 IEEE Aerospace Conference*. 4-Mar-2006. Cited pp. 1, 7.

[44] L. E. LaForge and J. W. G. Turner, "Multi-Processors by the Numbers: Mathematical Foundations of Spaceflight Grid Computing". *Proc., 2006 IEEE Aerospace Conf.* 4-Mar-2006. Cited pp. 1, 4, 5, 7, 8, 9, 12, 13, 14, 20; Figs 1, 2.

[45] T. Leighton and C. E. Leiserson, "Wafer-scale Integration of Systolic Arrays". *IEEE Transactions on Computers*. **C-34** (5), May, 1985, pp. 448 – 461. Cited p. 9.

[46] L. Lovász, "On the Ratio of Optimal Integral and Fractional Covers". *Discrete Mathematics*. **13** (4), 1975, pp. 383 – 390. Cited p. 7.

[47] C. M. Maunder and R. E. Tulloss, *The Test Access Port and Boundary Scan Architecture*. Los Alamitos, CA: IEEE Computer Society Press, 1990. Cited p. 9.

[48] E. F. Moore and C. E. Shannon, "Reliable Circuits Using Less Reliable Relays, Part I". Early, perhaps first, use of *quorum* on p. 202. *Journal of the Franklin Institute*. **262**, Sep-1956, pp. 191 – 208. Cited p. 3.

[49] O. Ore, *Theory of Graphs*. Providence: American Mathematical Society Publications, 1962. Cited p. 7.

[50] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall: Englewood Cliffs, NJ. 1982. Cited pp. 7, 8.

[51] F. Preparata, G. Metze, and R. Chien, "On the Connection Assignment Problem of Diagnosable Systems", *IEEE Transactions on Computers*. **EC-16** (6), Dec-1967, pp. 848 – 854. Cited pp. 7, 9.

[52] D. Renner, *OPNET Software Available Under U.S. GSA Schedules Program*. Press Release, OPNET, Incorporated. 19-Sep-2001. Cited p. 4.

[53] W. Rudin, *Functional Analysis*. New York: McGraw-Hill. 1973. Cited p. 3.

[54] E. R. Scheinerman, "Almost Sure Fault Tolerance in Random Graphs". *SIAM Journal of Computing*. **16** (6), Dec-1987, pp. 1124 – 1134. Cited p. 9.

[55] J. D. Ullman, *Computational Aspects of VLSI*. Rockville, MD: Computer Science Press. 1984. Cited p. 14.

[56] United States Missile Defense Agency, "Adaptable/Reconfigurable Distributed Spacecraft Processing". Department of Defense SBIR Topic MDA04-183. Aug-2004. Online at www.acq.osd.mil/sadbu/sbir/solicitations/sbir044/. Cited p. 4.

[57] United States Navy, "W Band, Real Time Wireless Network for Avionics Applications". Small Business Innovative Research (SBIR), Department of Defense SBIR Solicitation Topic N05-142. 1-Aug-2005. Online at http://www.dodsbir.net/solicitation/sbir053. Cited p. 7.

[58] E. Wilkeit, "Isometric Embeddings in Hamming Graphs". *Journal of Combinatorics, Series B*. **50**, 1990, pp. 179 – 197. Cited p. 16.

BIOGRAPHICAL SKETCHES

Laurence E. LaForge (IEEE Member) is President of the Right Stuff of Tahoe. Previously on faculty with Embry-Riddle Aeronautical University, he twice held NASA/ASEE Fellowships at the Jet Propulsion Laboratory, where he worked on bus fault tolerance and the *Deep Space 1* probe. He has been guest editor for the *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, and program chair for the IEEE International Conference on Innovative Systems in Silicon. His baccalaureate in mathematics is from the Massachusetts Institute of Technology; his PhD is from McGill University.

Jeffrey R. Moreland is a Digital Engineer with The Right Stuff of Tahoe. He develops the software shown in Figure 1, as well as the RightCardWare™ line of products for high-density barcode applications. Previously, he was a test engineer at Ametek Aerospace. His BS in Electrical Engineering is from the State University of New York at Binghamton.

M. Sami Fadali (Senior IEEE Member) earned a BS in Electrical Engineering from Cairo University in 1974, an MS from the Control Systems Center, UMIST, England, in 1977, and a Ph.D. from the University of Wyoming, in 1980. From 1983 to 1985, he was a Post Doctoral Fellow at Colorado State University. Since 1985, he has been on faculty at the University of Nevada, Reno, where he is a Professor of Electrical Engineering. His research interests include robust control, fault detection, and K-12 education. Professor Fadali is author for more than a hundred publications.