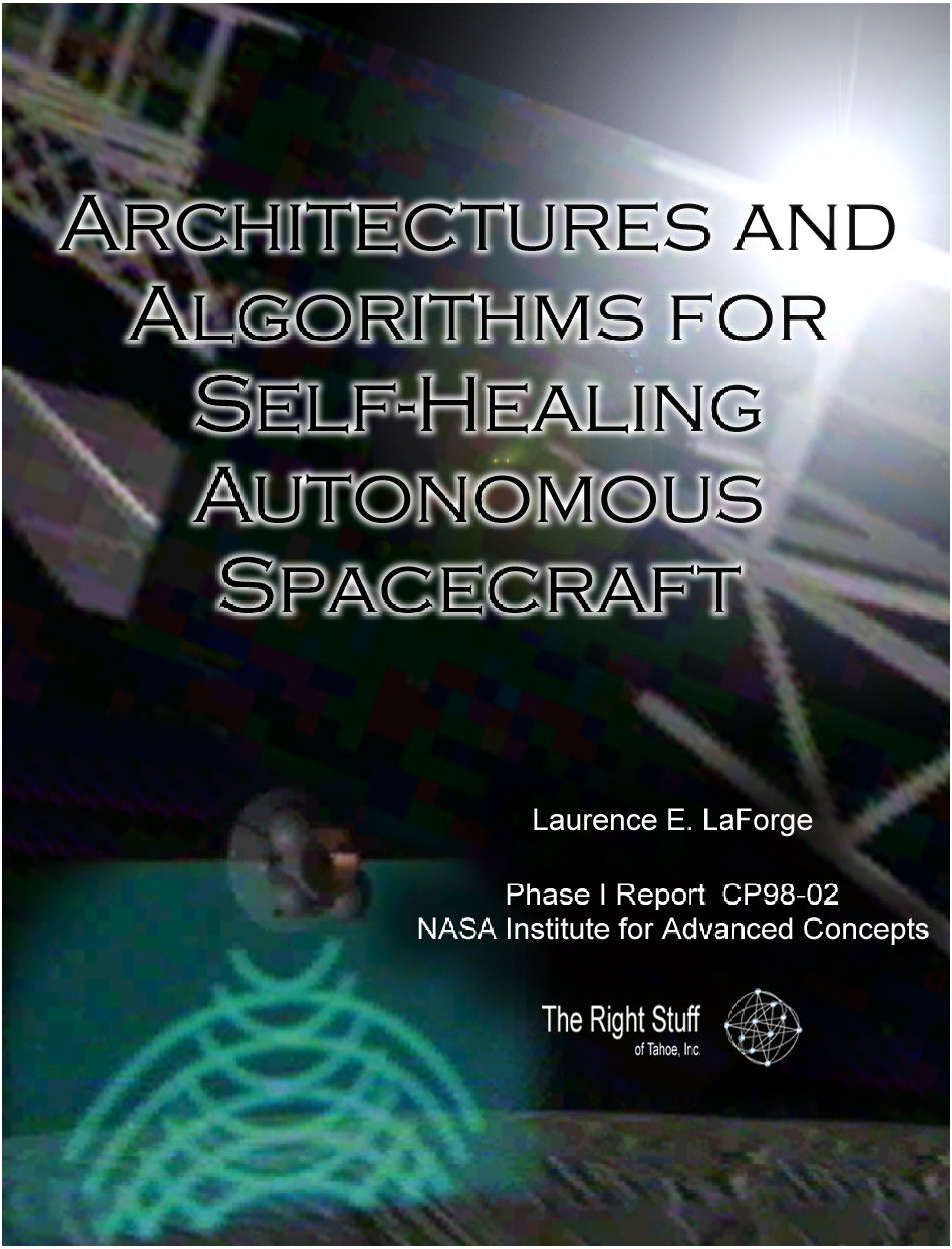


Architectures and Algorithms for Self-Healing Autonomous Spacecraft

Laurence E. LaForge

Phase I Report CP98-02
NASA Institute for Advanced Concepts

The Right Stuff of Tahoe, Inc.

Architectures and Algorithms for Self-Healing Autonomous Spacecraft

Laurence E. LaForge

Phase I Report CP98-02
NASA Institute for Advanced Concepts

The Right Stuff of Tahoe, Incorporated

9-Jan-2000
Revised 28-Feb-2000

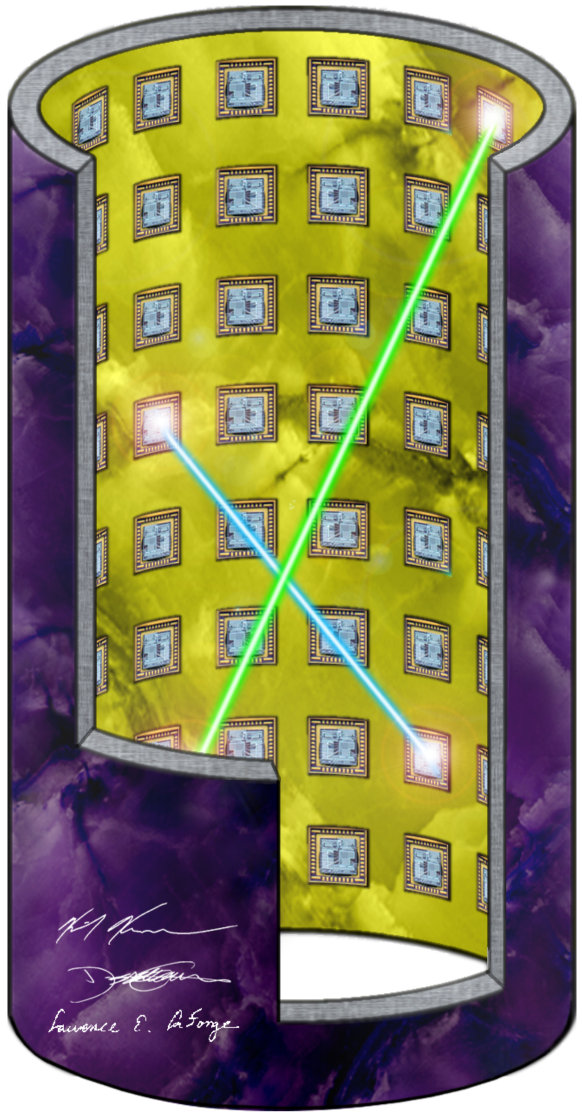


Figure 1: Self-healing *ProtoStar* multicomputer architecture, synthesized by our STAArchitecture software, implemented with optical interconnect and a cylindrical package.

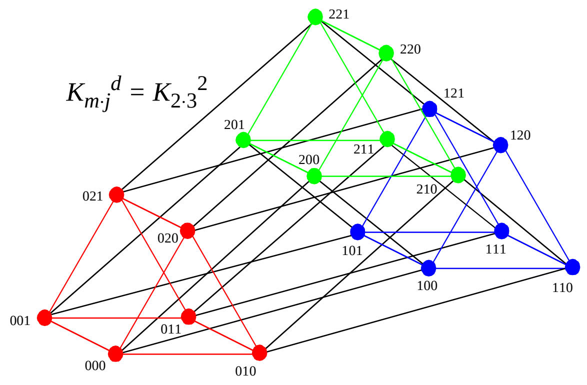


Figure 2: K-cube-connected edge exemplifying how our STAArchitecture software synthesizes a minimum-cost architecture with maximum fault tolerance and optimal performability.

Architectures and Algorithms for Self-Healing Autonomous Spacecraft

The Right Stuff of Tahoe, Inc.
The Right Place
3341 Adler Court
Reno, NV 89503-1263

Phase I Report CP98-02
NASA Institute for Advanced Concepts

Copyright © 2000

Laurence E. LaForge

Tel: (775) 322-5186
Fax: (775) 322-5182

Larry@The-Right-Stuff.com

Table of Contents

Cover background from [Milky Way 1999]. Cover design by Derek Carlson.

This revision reflects clarifications and corrections to versions issued 9, 12, and 30-Jan-2000.

Accompanying software. This report is accompanied by version 0.5 of STAArchitecture, a 32-bit Windows program described in Section 4. For operational and licensing information about STAArchitecture, refer to the readme.txt bundled with the online or CD ROM distribution kit. Also included in the distribution kit: astronomical-proportions.xls, a Microsoft Excel 97 spreadsheet that contains many of the calculations and charts featured herein. "STAArchitecture" and "Astronomical Proportions" are trademarks of The Right Stuff of Tahoe, Incorporated.[a]

Acknowledgments. At The Right Stuff of Tahoe, my co-investigators, Kirk F. Korver and Derek D. Carlson, wrote the STAArchitecture software, managed our office and accounts, performed literature searches, checked out mathematical models, and assisted with the production of this report. Myron Hecht of SoHAR elaborated ideas for self-healing software. Christopher R. Cassell[b] of Lockheed Martin lent his expert guidance in our calculations for Figure 5. Also of Lockheed Martin, Joe Marshall and David Meyer generously explained state-of-the-art radiation-hardened microcircuits, while Homayoun Malek provided insight into differences and similarities between starship applications and the Theater High-Altitude Area Defense program (Thaad) [Lewis and Postol 1997], [Stein and Little 1997]. For the Hertzsprung-Russell diagram of Figure 3, Roger A. Freedman of the University of California at Santa Barbara graciously secured permission from W. H. Freeman and Company to reproduce Figure 19-13 of *Universe* [Kaufmann and Freedman 1999], the wonderful astronomy text he wrote with William J. Kaufmann III.

I am particularly indebted to Sarah A. Gavit for sharing the results of her team's work on the JPL Interstellar Probe. Her colleagues at NASA facilities – Ames Research Center, Glenn Research Center, the Jet Propulsion Laboratory, Kennedy Space Center, Langley Research Center, Marshall Space Flight Center, and Washington Headquarters – provided additional valuable material, both written and spoken. Though generally credited among the references of Appendix A, I would like to reiterate my appreciation to: Leon Alkalai, John R. Anderson, Robert C. Barry, Douglas E. Bernard, N. Talbot Brady, W. H. Bryant, Savio N. Chau, John Cole, James M. Dumoulin, Daniel L. Dvorak, David J Eisenman, Daniel B. Eldred, Daniel E. Erickson, Martin S. Feather, Dwight A. Geer, Peter R. Glück, Daniel S. Goldin, Cecilia N. Guiar, Keith D. Goodfellow, Donald J. Hunter, Sanford M. Krasner, Vincent R. Lalli, David H. Lehman, Michael R. Lowry, Marc G. Millis, David Morrison, Brian K. Muirhead, Allen P. Nikora, Peter Norvig, John C. Peterson, Robert D. Rasmussen, Allan L. Sacks, Joseph L. Savino, Francis L. Schneider, Greg Schmidt, Burton C. Sigal, Carl N. Steiner, Samuel L. Venneri, and David F. Woerner.

I would also like to express my appreciation to colleagues at Embry-Riddle Aeronautical University. In the College of Career Education: Dr. William Getter, Dean of Academics. At the Hunt Memorial Library: Ann Cash, Ellen Dewkett, Kathleen Chumley, Maria O' Brien, and Christine Poucher.

Thank you, one and all!

a. The logo in the corners of these pages is also a trademark of The Right Stuff of Tahoe, Incorporated. It depicts the second smallest member of a class of graphs, discovered in the early 1990's to be almost surely diagnosable at constant test redundancy [LaForge et al 1993].

b. Representing himself.

1. Executive Summary

Imagine interstellar missions lasting five times your lifespan … Our spacecraft orbits Alpha Centauri, decides which moon of the second planet to explore, plans how to sample soil from that moon's craters, and lands micro-rovers that burrow beneath the craters' surface. Figure 3 illustrates how our work dovetails with two research areas deemed by NASA as critical to such interstellar missions. From the standpoint of self-healing architectures and algorithms, this report describes opportunities and challenges that are key to launching, within the next forty years, spacecraft that are truly autonomous.

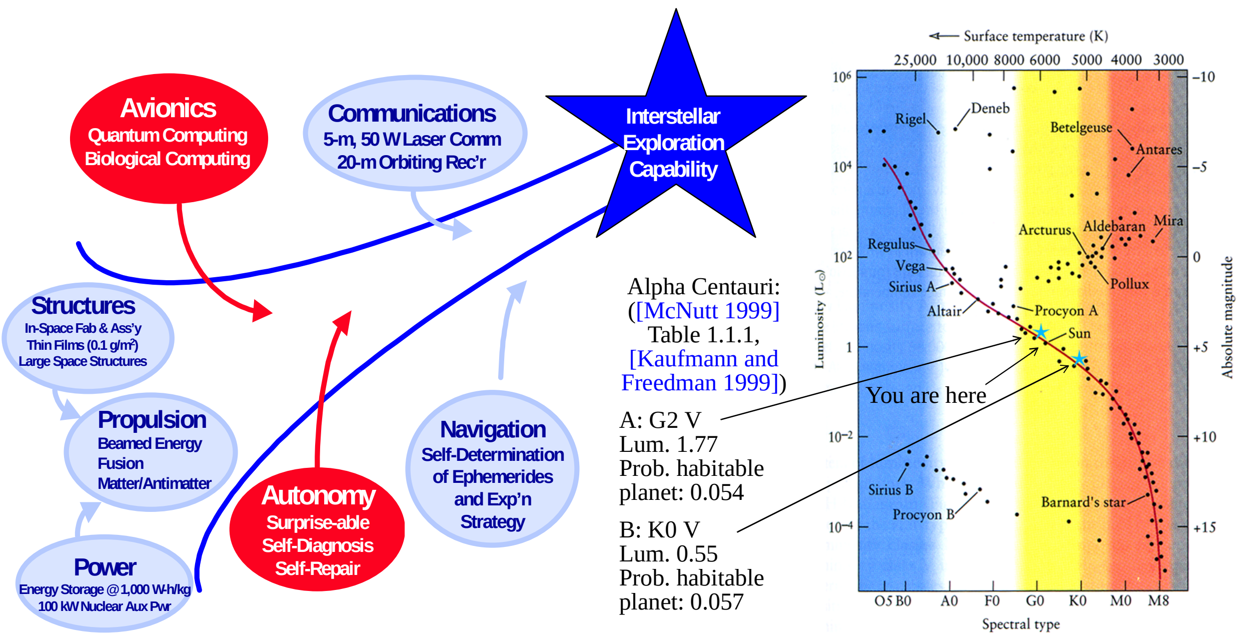


Figure 3: Enablers and neighboring destinations for interstellar missions. Left side from [Gavit 1999].

This report is news that you can use. Apart from Sections 4.3 through 4.5, we have written for readers versed in one or more technical disciplines, but who are not necessarily specialists in adaptive and fault tolerant systems. Sections 2 through 4 flesh out the Phase I goals, findings, and conclusions synopsized by Table 1. In Section 4 we apply our core competence of analytic modeling, along with our background in experimental and simulated systems, to a problem domain that is both broad (Sections 2 and 3) and deep. Among our major contributions we feature the bibliography of Appendix A, and highly recommend the 170 references therein to investigators in search of greater breadth and depth. Stemming from our Phase I work, for example, our citations include

- a journal article submitted ([LaForge 1999 Trans. Reliability])
- three peer-reviewed conference papers accepted for publication ([LaForge and Korver 2000 MTAD], [LaForge 2000 Starship Avionics], [LaForge and Korver 2000 Graph Fault Tolerance])
- a NASA technical report update ([LaForge 1999 JPL D-16485])

with the latter two contributing theorems whose consequences for hypercube-related structures are both encouraging and provocative (*cf.* Sections 4.3 through 4.5). However, self-healing autonomous spacecraft will not just magically sprout from evolutionary improvements in earth-based software and hardware [Goldin et al 1998]. Using this report as a springboard, Section 5 projects avenues for serious development of self-healing autonomous starship multicomputers, such as that exemplified in Figure 1. Whether or not you have a penchant for mathematical rigor, we hope that you find this report to be informative, interesting, and even a touch inspirational.

Details	Goal	Thematic Highlights
Sec. 2	Profile missions requiring revolutionary survivability and performance.	Substantive answers to NASA's Grand Challenges [NIAC 1999 Scope], particularly exploration beyond the solar system, require unmanned spacecraft that can operate between one century and one millennium, intelligently and without earth-based tactical control.
Sec. 3	Characterize computational avionics and software that will enable missions profiled in 2.	Key to success: hardware/software co-design capable of maintaining low-level health of computational resources, while simultaneously facilitating high-level adaptation, over time and in the presence of a wide range of onboard and external conditions.
Sec. 4	Advance state-of-the-art for mission-critical architectures and algorithms.	Point-to-point interconnection is best bet for maintaining a working quorum, excising problematic subsystems. In a ratioed asymptotic sense, clique-based cubes are superior to traditional hypercubes.
Sec. 4	Demonstrate, communicate, and automate application of advances in the state-of-the-art	Reflecting breakthroughs in the theory of emerging properties, new design tools are essential to self-healing autonomy. STAArchitecture, a CAD program of our own creation, exemplifies such a tool in the domain of graph-theoretic interconnection.
Sec. 3 Table 3 Sec. 5	Survey and forecast technologies appropriate for self-healing autonomy.	Akin to the *International Technology Roadmap for Semiconductors* [SRC 1998], [Geppert 1999], self-healing autonomy is well-served by a *Starchart* that lists, in an integrated fashion, known, desired, and possible characteristics of materials, circuits, manufacturing, and software engineering. This report serves to begin such a *Starchart*.

Table 1: Framework established by Phase I of this work.

2. Starships Require Revolutionary Autonomy, Survivability, and Performance

Perhaps more than any other factor, *distance* provides the impetus for spacecraft autonomy ([Goldsmith 1999], Chap. 9). Our knowledge of a spacecraft's environment diminishes with increasing distance from earth (but this makes the mission more interesting). Distance dictates *communication latency*: to arrive at a probe near Pluto,[1] for example, a radio signal travels for more than five hours – entirely too long for earth-bound controllers to tactically adapt the spacecraft to changes in environment.

Distance also dominates properties of self-healing avionics and software. Most prominently, the *time for a probe to complete its mission* is bounded from below by the length of the journey divided by the speed provided by our best propulsion. For either hardware or software, the cumulative probability of failure increases with the time of operation.[2] Mission duration is therefore a major challenge for continuously available avionics and software. These observations underscore why revolutions in self-healing autonomy are most needed, and why they are most likely to be devised, for long distance missions of long duration.

Refer to Table 2. As a framework for the goals, requirements, and properties presented in Section 3, it is worthwhile to consider three interstellar missions envisioned for the next ten to forty years:[3]

i) The Interstellar Probe ("JPL ISP") being developed at the Jet Propulsion Laboratory [Gavit 1999]
ii) The Interstellar Precursor Mission ("McNutt IPM") proposed by Ralph McNutt at Johns Hopkins University [McNutt et al 1997], [McNutt 1999]
iii) *Santa Maria*, a mission of our own conception [LaForge et al 1999]

1. *Pluto-Kuiper Express* will explore this region, 40 Astronomical Units (AU) from earth [Alkalai and Tai 1998].
2. *Cumulative failure probability* equals one minus the reliability ([Siewiorek and Swarz 1982] p. 31).
3. Many of our assumptions and results apply as well to missions within the solar system ([Cassanova 1999] p. 7).

Mission → Characteristic ↓	Jet Propulsion Laboratory Interstellar Probe (JPL ISP) [Gavit 1999]	Johns Hopkins University (McNutt IPM) [McNutt et al 1997], [McNutt 1999]	The Right Stuff of Tahoe: *Santa Maria* [LaForge et al 1999]
Destination	nose of heliosphere	beyond nose of heliosphere	Alpha Centauri
Distance (AU)	200	1000	278256
Trajectory	ecliptic, with solar swingby	ecliptic, with Jovian / solar swingby	direct
Propulsion	solar sail	Orion class nuclear	fusion/antimatter, $I_{sp} > 10^5$ sec
Launch (Gregorian date)	2010		August 2, 2022
Duration (years)	15	50	between 100 and 600
Payload mass (kg)	100 (science instr: 25)	50 (science instr: 10)	1000 (science instr: 250)
Power (continuous, watts)	20	15	100
Downlink capacity bps (bits per second) at 100 AU	500	1000	10000

Table 2: Baseline missions driving requirements for autonomy, survivability, and performance.

In addition to distance and mission duration, Table 2 lists three quantities that are directly pertinent to our research. In rough order of importance, these are *payload mass*, *average available power*, and *communication capacity*. By way of reference, each of the two *Voyager* spacecraft has a mass of 800 kilograms, generates 470 watts of nominal power (degraded to 340 watts at present), and downlinks data to Deep Space Network (DSN) 34 meter antennae at 160 bps [Wade 1999], [JPL 1999 Voyager]. [Guiar 1998] details facts and figures about the planned *Europa Orbiter* and *Pluto-Kuiper Express*.

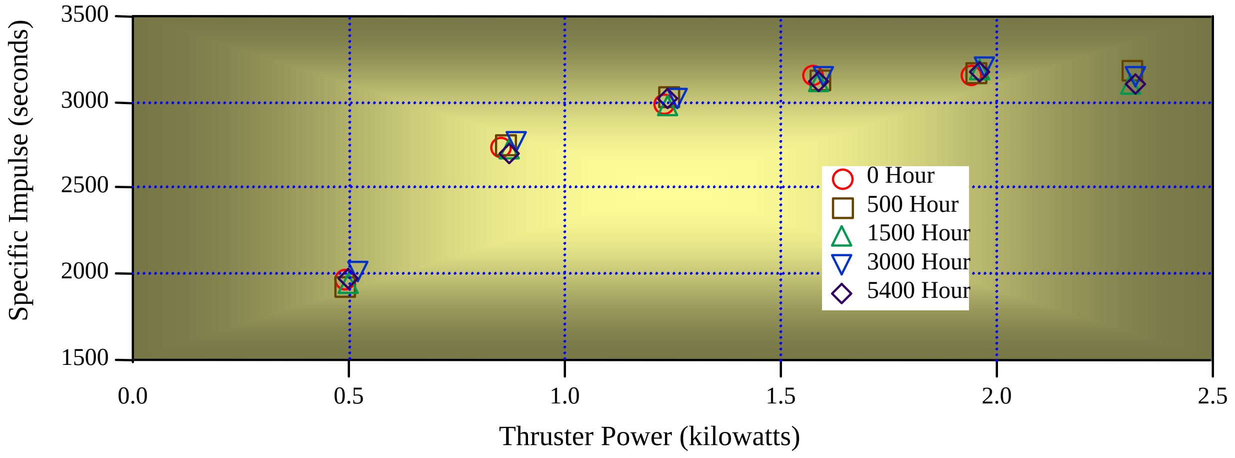
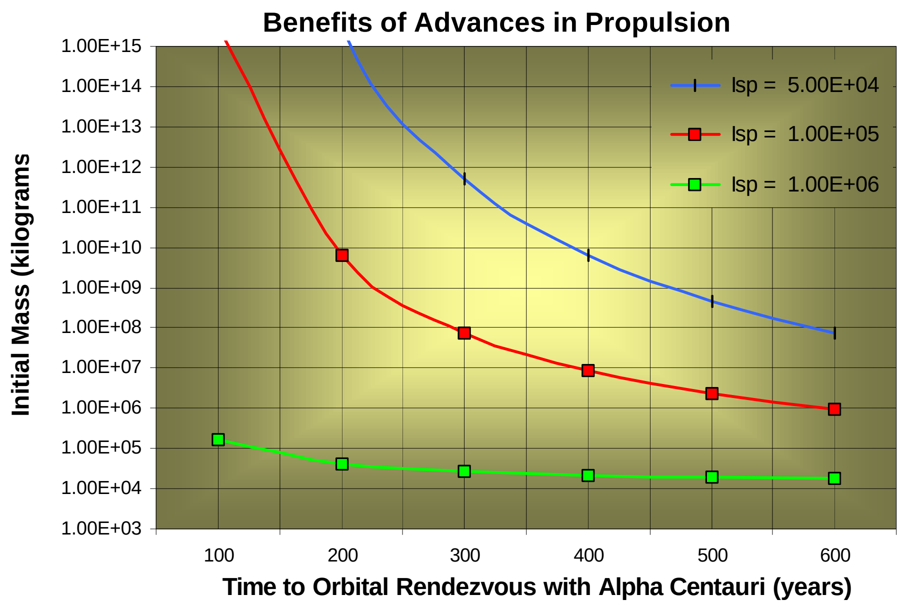


Figure 4: *Deep Space 1* ion engine I_{sp} as measured at the Jet Propulsion Laboratory [Anderson 1999].

Santa Maria is qualitatively different from either the JPL ISP or the McNutt IPM. Refer to Figure 4. With a specific impulse (I_{sp}) of between 1950 and 3150 seconds, the *Deep Space 1* (*DS1*) ion engine represents the state-of-the-art in propulsive efficiency.[4] By contrast, conversations with our fellow NIAC researchers ([Kammash 1999], [LaPointe 1999], [Slough 1999]) suggest that order-of-magnitude advances in propulsion may well place *Santa Maria* within the realm of what we can achieve, within the next quarter century [Millis 1997].[5] Making use of the rocket equation [Larson and Wertz 1995], Figure 5 plots tradeoffs among *Santa Maria* mission duration, initial mass, and propulsive efficiency.[6] Not shown is the considerable reduction in resources required for a fly-by of (as opposed to orbital insertion around) the Alpha Centauri ternary ([Goldsmith 1999], p. 168).[7] However, even with a fly-by that uses the most efficient propulsion forecasted, the initial mass of *Santa Maria* is likely to exceed that of several Saturn/Apollo vehicles.[8] In light of this, it makes sense to assume that *Santa Maria* will be assembled and launched from a space-born platform, such as the International Space Station or an earth-moon Lagrange point.[9] The attendant logistics point to a need for revolutionary new levels of *modularity* in avionics components.



Application of the nonrelativistic rocket equation:

$$m_p = m_f\left(e^{\frac{\Delta V}{g \cdot I_{sp}}} - 1\right)$$

payload = 10^3 kg
engine = 10^4 kg

Constant acceleration/decceleration, symmetric about half-way point.

Structure and tank overhead = 3% of propellant required to deliver engine and payload; incremental shedding of tankage.

Figure 5: Revolutionary breakthroughs in propulsion will make it feasible to launch a probe to Alpha Centauri within the next twenty to forty years. However, even with a hundredfold improvement over the efficiency of the *Deep Space 1* ion engine, the journey will take more than a century to complete.

We conclude this section by addressing a question posed by Professor Webster Cash ([NIAC 1999 Agenda], [Cash 1999]). Concerning the timing of a launch to Alpha Centauri within the next four decades:

Would later generations not send starships, featuring more capable propulsive systems, that would overtake *Santa Maria*?

Although the scenario above is certainly possible, the history of space exploration suggests otherwise. For example, the last people to reach the moon left its surface on 14-Dec-1972 [Dumullin 1999]; in spite of technological advances over the last twenty-eight years, however, we have yet to send another person on a lunar mission. As a second example, since launching *Voyager 1* and *Voyager 2* in late summer of 1977 [JPL 1999 Voyager], we have yet to send another probe to the edge of the solar system. By maintaining a

4. Other aspects of the *DS1* ion engine are summarized in [Braham 1999].
5. McNutt does not believe we can achieve I_{sp}'s of 10^5 seconds within the next forty years ([McNutt 1999] p. 6).
6. Proposed launch date: 530th anniversary of Christopher Columbus sailing from Palos, Spain [Pickering 1999].
7. Standard texts treat Alpha Centauri as a binary of stars of type G2 V and K0 V [Kaufmann and Freedman 1999].
8. Approximate mass of a fully loaded three-stage Saturn V with Apollo payload: 2.9×10^6 kg [Richard 1999].
9. Assembly of *Santa Maria* at the International Space Station would bolster synergism between manned and unmanned programs [Oberg 1998]. JPL proposes an earth-based Delta II rocket launch of ISP [Gavit 1999].

wait-and-see strategy for improvements in propulsion, moreover, we might never launch a mission to Alpha Centauri. Consistent with NIAC objectives [NIAC 1999 Scope], this report presumes, and furthermore advocates, charting a course for Alpha Centauri at the earliest feasible opportunity. The remaining sections chart a course of feasible opportunities for self-healing computational avionics and software.

Goal/Requirement/Property	**Priorities for Success: ★ Stars of the Main Sequence** **Section 3 at a Glance** **In-depth Research, Phase I:** **Section 4 of this report** **STAArchitecture CAD Tool, Phase I:** **Section 4 of this report**	**Emphasis**				
		Avionics	**Autonomous Intelligence**	**Self-Healing Archs. & Algs.**	**CAD Tools**	**Design Processes**
3.1	Payload: roving astronomer-on-a-computer.		★			
3.2	Avionics targeted to support software that can learn and adapt on its own, occasionally guided via communication (albeit highly latent) with earth.	★	★			
3.3	More's Law:[12] Autonomous Intelligence (AI) demands more computing power.		★			
3.4	Experimentally verified models quantify relations among processing power, low-level tasks, and AI. 10^{15} ops/sec-kg provisional performance objective.		★			
3.5	Four domains of knowledge and control: a) scientific; b) communication; c) navigation; d) self-healing starship control and maintenance.		★			
3.6	**Tolerance to a constant proportion of faulty computational nodes.**			★		
3.7	**Self-configuring, uniform computational nodes maintain connectivity among healthy nodes, disconnect healthy from faulty nodes.**			★		
3.8	**Computational nodes identify faults via MTAD: mutual test and diagnosis.**			★		
3.9	**Computational nodes govern their own switches for connecting to other nodes.**	★		★		
3.10	Switch technologies biased against stuck-closed faults.	★				
3.11	Computational nodes connected by three-dimensional technology that mitigates, in order of priority: a) short circuits, b) mass, and c) signal delay.	★				
3.12	Shielding and hardening to 10^4 Mrad(Si)	★				
3.13	Reuse only components that match design requirements.	★	★			
3.14	**Teams embrace and exploit a new generation of CAD tools and processes.**				★	★
3.15	Avionics and software designed for high yield as well as high reliability.	★	★			★
3.16	Instrument and measure software failures and faults at quantifiable levels commensurate with physical measurements of avionics.				★	★
3.17	Administrators and managers actively promote, and participate in, development.					★

Table 3: *Starchart* genesis: avionics and software co-design, enabling characteristics at a glance. This table, along with the bulk of Sections 2, 3, and 5, will be published in [LaForge 2000 Starship Avionics].

3. *Starchart* Genesis: Enabling Characteristics of Starship Software and Avionics

From the perspective of architectures and algorithms, Table 3 proffers priorities about what we should build and, furthermore, how we should go about building it. Though by no means exhaustive, characteristics 3.1 through 3.17 embrace the space of *software/hardware co-design* for self-healing autonomous starships. With respect to this design space, the descriptions, examples, and references cited in this section establish a central role for self-healing architectures and algorithms (requirements 3.6 through 3.9, Section 4) as well as related tools (requirement 3.14, Section 4) for computer aided design (CAD). We begin with a position put forth by Dr. Daniel Dvorak and Dr. Richard Doyle at JPL [Dvorak and Doyle 1998], and by Professor Sandra M. Faber of the University of California at Santa Cruz [Milky Way 1999]:

3.1 Starship payloads will be autonomous, intelligent, robots: roving astronomers-on-a-computer.[10]

As basic as 3.1 may be, it sets the stage for the computational tasks that avionics and software must perform. While press reports may have exaggerated the success of the *Deep Space 1* Remote Agent ([JPL DS1 Press Release], [Foust 1999]), real starship autonomy is within the twenty-year horizon of software engineering. JPL's X2000 Mission Data System is being built of Goal Achieving Modules (GAMS), an evolutionary successor to the Remote Agent [Dvorak 1998]. By any stretch of the imagination, therefore, starship avionics must provide the raw computational horsepower for successors to the Remote Agent.

Software is the only thing that we can add to a payload, once our starship has ventured into deep space; software provides a final degree of freedom in starship design and implementation. From this simple observation we see that a very real opportunity (and perhaps the best opportunity) for maximizing autonomy is to ensure the starship software does not spend the time to destination spinning in an idle loop, but rather "grows up", in a fashion not unlike the way hominoids mature to adulthood:

3.2 Starship avionics will support software that can learn and adapt on its own, occasionally guided via communication (albeit highly latent) with earth.

As with other high-end applications, More's Law applies:

3.3 Autonomous intelligence (AI)[11] will always demand more computing power, more memory, and more communications capacity.[12]

For any *given* level of AI, we can ask how much computing power will be needed to satisfy requirements 3.2 and 3.3. The open-ended nature of this requirement (what does it mean for software to learn and adapt on its own?) renders such an assessment difficult. Refer to Figure 6. Our Phase I technical proposal targeted a machine capable of retiring 10^{15} operations per second [LaForge 1999 NIAC Phase I Proposal], and this appears to be achievable. Largely due to the dearth of models and experiments for characterizing AI workloads [Dvorak 1999], however, we really do not know if such performance is too little, too much, or about right. Section 5 identifies a number of tasks critical to workload characterization, at a level of fidelity that will facilitate the design of self-healing architectures and algorithms. In the interim:

3.4 Successful design of self-healing autonomy will require experimentally verified models that quantify the relations among processing power, low-level tasks, and AI.[13] 10^{15} operations per second per kilogram is a provisional performance objective.

10. Even optimists doubt the feasibility of human interstellar exploration within the next forty years [Nordley 1998].
11. AI = *autonomous* (not artificial) intelligence: intentional coining of new words for an old, artificial acronym.
12. Requirement 3.3 articulates More's Law. Moore's Law states that our capacity for packing circuit devices or information storage increases exponentially (alternatively, the price per circuit function or bit of storage decreases exponentially), at between 2% and 4% per month [Schaller 1997], [Economist 1997], [Hamilton 1999].
13. One possible application for experimentally quantifying such a workload: IVHM, an AI system for self-diagnosis of remote agents, spaceliners, and rotorcraft; under development at NASA Ames Research Center [Norvig 1999].

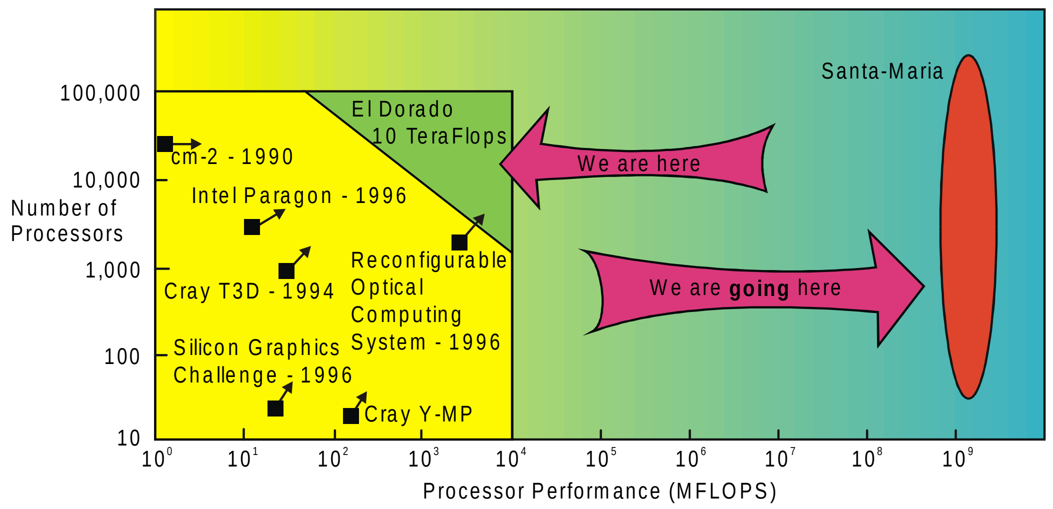


Figure 6: Trends in high-end processing power. Adapted from [Szymanski and Supmaonchai 1996].

Notwithstanding our inability to quantify the computational demands imposed by an autonomously intelligent workload, we can *qualify* some of the salient characteristics of such a system. For example, what kind of knowledge will our starship need to possess, and what kinds of things will it have to do in response to its environment? Broadly speaking:

3.5 The combination of knowledge and control of actions (based on knowledge and input) can be divided into four domains: a) scientific; b) communication; c) navigation;[14] d) starship control and maintenance, with the latter including self-healing architectures and algorithms for computational resources.

From a strategic perspective, science is the *raison d'être* for a starship: the navigation, communication, and instrument control serve as means for conducting and reporting the outcome of science experiments. From a tactical point of view, the order of priority is reversed: the starship's computational health is necessary to successfully determine location, trajectory, and control; accurate navigation is necessary to positioning spacecraft to carry out science experiments; the value of these experiments hinges on the starship's ability to communicate their outcomes to us. While our Phase I effort focuses on 3.5d, a comprehensive design of hardware and software will take full account of 3.5a, b, and c. These domains are traditionally addressed by formulating specifications for onboard instrumentation. Refer to Table 4. While both the JPL ISP and the McNutt IPM feature manifests for onboard scientific instruments ([Gavit 1999], [McNutt 1999]), neither of these missions appears to have developed an analogous list for navigation and communication.[15]

The genesis of starship instrumentation may well result from a combination of evolution and revolution. For example, new discoveries of planets outside the solar system,[16] perhaps with the aid of x-ray interferometers like that proposed by Professor Webster Cash [Cash 1999 X-File], could shift our investigative priorities to questions about astrobiology [Morrison and Schmidt 1999]. In the case of *Santa Maria*, it is beyond our Phase I scope to draft a list of instruments – scientific, navigational, communication, or otherwise. However, Section 5 identifies collaborative tasks for composing, at an operational level of fidelity, a manifest for the *Santa Maria* payload. We anticipate a diverse yet coordinated suite of instruments, reflecting contributions from specialists in academia, industry, and government.

14. We include control of propulsive actuators as falling properly within the domain of navigation.
15. Although absent such a list for navigation and communication, a substantial amount of the McNutt IPM Phase I report is devoted to these aspects ([McNutt 1999], Sec. 2.6, 2.8).
16. Astronomers believe they have discovered 28 planets outside of the solar system; with masses on a Jovian scale, some of these may be brown dwarfs [Recer 1999], [Butler et al 1999].

JPL ISP		**Science instrument**	**McNutt IPM**	
Yes		Magnetometer	**3.0 kg**	**0.5 watts**
Yes		Plasma wave / radio detector	**1.5 kg**	**2.5 watts**
Yes		Dust sensor		
Yes		Detector for cosmic rays, hydrogen, helium, electrons and positrons	**1.0 kg**	**1.5 watts**
Yes		Galactic cosmic ray composition analyzer		
Yes		Solar wind / interstellar plasma / electron sensor		
Yes		Pickup and interstellar ion composition analyzer		
Yes		Interstellar neutrals detector	**2.0 kg**	**2.0 watts**
Yes		Suprathermal ion / electron sensor		
Yes		Energetic neutral atom imager		
Yes		Infrared imager	**1.5 kg**	**1.5 watts**
Yes		Ultraviolet photometer	**No**	
No		Lyman-α imager	**1.0 kg**	**2.0 watts**
25 kg	**20 watts**	**All Science Instruments**	**10 kg**	**10 watts**

Table 4: Payload manifests instrumental to starship science experiments [Gavit 1999], [McNutt 1999].

Despite gaps in our understanding of the implications of the more strategic domains of 3.5, we can characterize many facets of computational avionics and software for interstellar missions. In particular, *hardware dependability* is at least as important as computational power, and in this respect we can bracket both what is required and what is possible. Contemporary spacecraft avionics are specified to tolerate one fault, perhaps two, and this limited tolerance is mandated without regard to either the component failure probability or the number of components [Barry 1998], [Guiar 1998]. On the other hand, estimates for the mean time to failure (MTTF) of discrete components range from 100 to 1000 years [Avizienis 1999], [Virgras], [Harris]. In light of this, and taking into account burgeoning levels of system complexity:

3.6 Self-healing autonomous starships will tolerate a *constant proportion* of faulty computational nodes.[17]

Reflecting the needs generated by century-long interstellar missions, Professor Algirdas Avizienis of the University of California at Los Angeles has outlined both the past and the future of fault tolerant computational avionics [Avizienis 1999]. As to the latter, starships face three categories of threats: i) latent design faults that remain onboard; ii) components wearing out; iii) transient faults, single or burst. To this list we would like to add iv) permanent faults that arise during the mission, regardless of source (*e.g.*, radiation, thermal cycling, metal migration, latchup)[18].

17. This stands in contrast to contemporary missions with requirements for tolerance to one or (in the case of STS launches) two faults [Guiar 1998]. Proportions can be estimated by integrating the failure probability density over mission duration. For example, assuming that every part fails according to a negative exponential distribution with identical failure rates then, for a mission duration equal to the MTTF, $1/e \cong 36\%$ of the components will have failed.

18. Category (iv) is in fact mentioned elsewhere in [Avizienis 1999], apart from (i) through (iii). Category (iv) overlaps with, but is not a necessary consequence of, (iii). For example, radiation or shock induced failure is arguably not the same as a component "wearing out".

The immune system paradigm proposed by [Avizienis 1999], and independently set forth by SoHaR President Myron Hecht [LaForge et al 1999], appears to be a promising approach for addressing threats (i) through (iv). With all respect, however, the DiSTARS implementation advocated by Avizienis is not the most prudent use of redundancy. In particular, static, hardware-level distinction of computational nodes (as *master*, *controlling*, or *defensive*) is contrary to much of what we have learned in the three decades since the JPL STAR computer was designed [Avizienis 1967], [Hecht and Fiorentino 1988].

More precisely, dedicated sparing (such as that suggested for DiSTARS) tends to be more costly than local sparing, but local sparing is generally more costly than uniform sparing. To cite a well-understood example: at constant proportion of failed elements, the (unfortunately) prevalent method of using dedicated spare rows and columns to boost the dependability of arrays (be they memory, register files, or cache lookup tables) requires redundancy that is *exponential* in the square root of the size of the array [LaForge 1999 Trans. Reliability]. Refer to Figures 7 and 8. Under the same probabilistic fault model, *local sparing* of array elements (known as *cross-strapping* by spacecraft designers at JPL, and depicted in Figure 2.7.1 of [McNutt 1999]) delivers the same dependability, but in this case the redundancy is *logarithmic* in the array size [LaForge 1999 Trans. Computers].[19] Under a VLSI layout model, moreover, uniform sparing costs less than either of these two approaches, with *log log* redundancy for two-dimensional arrays and constant redundancy and wirelength in the one-dimensional case [Leighton and Leiserson 1985].

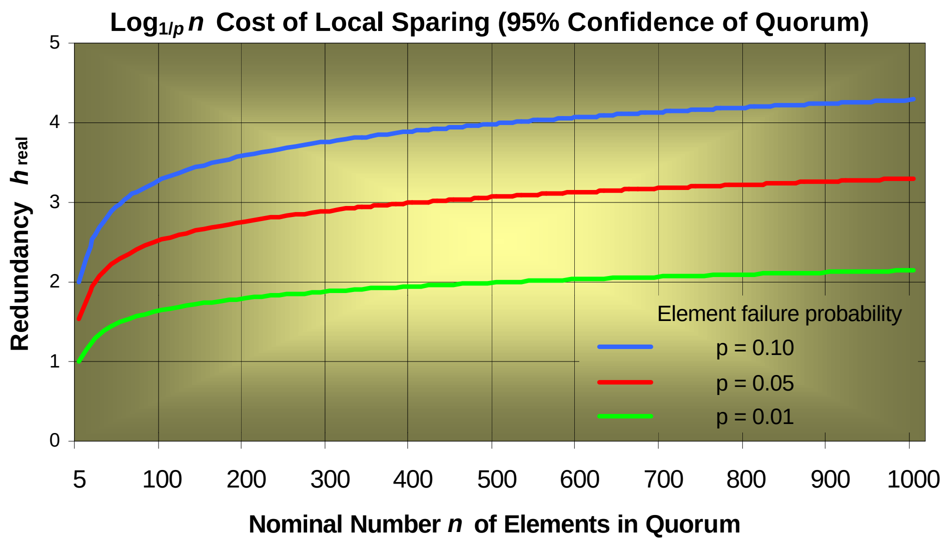


Figure 7:

How many spares to build when employing cross-strapped redundancy? Though these curves answer this fundamental question,[19] it appears that they (as well as the curves of Figure 8) have yet to be incorporated into CAD tools and processes for electronics design.

19. The logarithmic cost of local sparing is a fundamental analytic result, and perhaps warrants spelling out in detail. Suppose that our nominal quorum requires n working components, that components fail with Bernoulli probability p, and that we replace each component by a block containing h copies of the respective component (h is the *discrete ratioed component redundancy*). Cross-strap the components of any given block with all of the components in each adjacent block, where block adjacency in the redundant architecture is isomorphic to component adjacency in the nominal architecture. Denote by $Y = (1-p^h)^n$ our required probability that each of the n nominal components is represented by a working component (Y is the *configuration coverage*). Let c be any real value greater than 1 and less than $Y^{1/n}$. The minimum redundancy h that meets our mission requirements equals the least integer no less than h_{real}:

$$\log_{1/p} n - \log_{1/p} \ln \frac{1}{Y} - \log_{1/p} \frac{2}{c+1} > h_{\text{real}} > \log_{1/p} n - \log_{1/p} \ln \frac{1}{Y}$$, which converges quickly to the righthand side.

The above expression was first published in [LaForge 1994], with proof relying on bounded Taylor series. Independently, [Leighton and Leiserson 1985] note that the order of magnitude is logarithmic in n. [LaForge 1999 Trans. Computers] generalizes to a multivariate fault model that includes switches stuck open or closed.

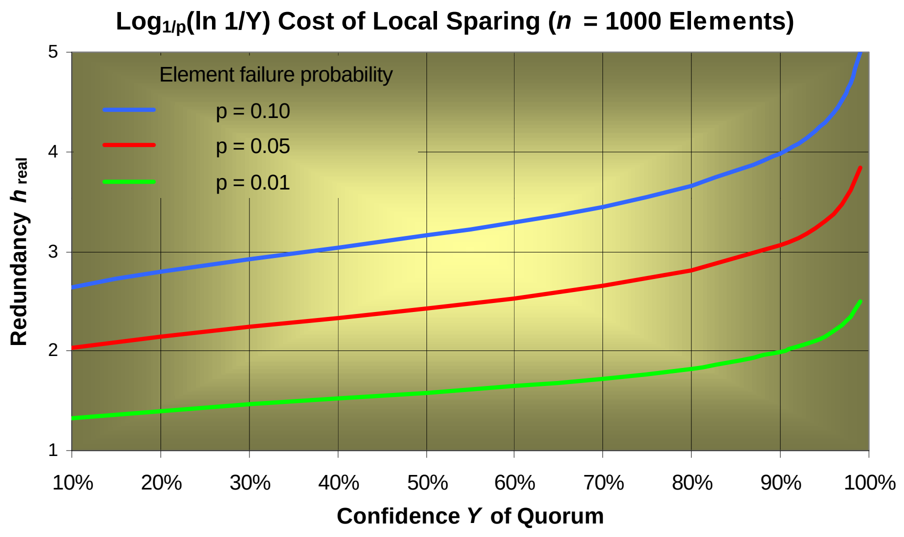


Figure 8: Optimal number of cross-strapped spares to build, as a function of quorum survivability *Y*.

More generally, by relaxing constraints on the shape of our target architecture (*e.g.*, not insisting on an array), while at the same time maintaining uniformity among computational nodes, we can achieve highly survivable systems whose redundancy is (a best possible) *constant*. For illustration refer to Figures 1 and 2. Reflecting this, and by contrast to recent FAT-tree experiments with the Teramac supercomputer [Clark 1998], [Culbertson et al 1997], [Culbertson et al 1996], Section 5 outlines tasks for creating a family of optimally redundant starship multi-computer whose design devolves from a simple requirement:

3.7 Computational starship avionics will consist of self-configuring, uniform computational nodes that maintain connectivity among healthy nodes, and that disconnect healthy from faulty nodes.

Such a collection of healthy nodes constitutes a *quorum*.[20] Since cooperative computation requires that healthy nodes communicate (and furthermore that they restrict communication with faulty nodes), 3.7 is a fundamental criterion for self-healing autonomy. We may (and generally will) augment 3.7 with performance requirements for minimizing latency or maximizing throughput. Resurgent interest in combining dependability with computational speed has sparked a relatively new area of research: *performability* [Haverkort and Niemegeers 1996]. Section 4 explains our Phase I contributions to performability theory, with novel emphasis on structure instead of traditional Markov chains [Nabli and Sericola 1996].

Consistent with requirements set forth by [Avizienis 1999]:

3.8 Computational nodes aboard starships will identify faults via MTAD: mutual test and diagnosis.

The need for *distributed* MTAD algorithms follows by observing that every node is subject to failure. Figure 9 depicts the relation between 3.7 and 3.8. As is the case with requirement 3.7, MTAD is essential to self-healing autonomy [LaForge and Korver 2000 MTAD]. However, MTAD represents a radical departure from traditional, centralized spacecraft fault protection ([Chau 1998 PDR], [LaForge 1999 JPL D-16485] pp. 58-70), and considerable experimental work remains to be done in order to best implement MTAD. In [LaForge and Korver 2000 MTAD] we describe how to go about performing these experiments.

20. This use of *quorum* appears to have originated with research at Bell Laboratories [Moore and Shannon 1956].

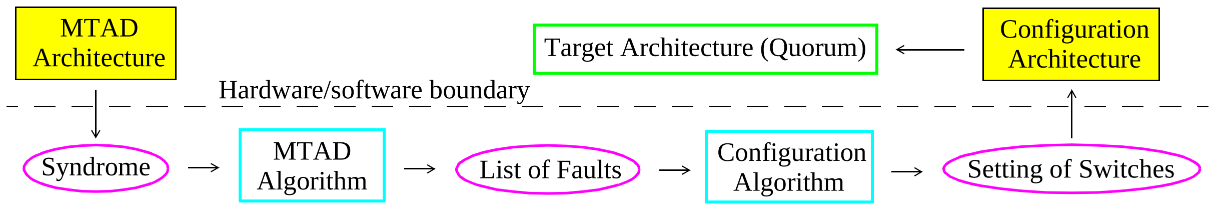


Figure 9: Diagnosis and configuration for self-healing autonomy: architectures versus algorithms.

Requirements 3.7 and 3.8 speak to properties for switching and interconnect:

3.9 Computational nodes aboard a starship will govern their own switches for connecting to other nodes.

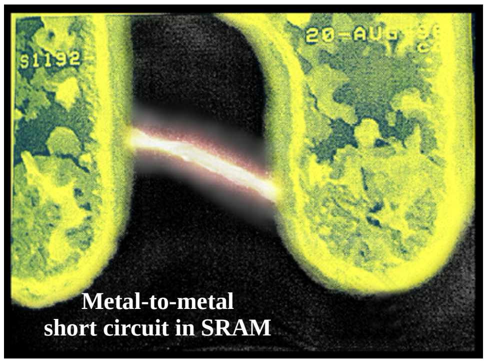
3.10 Starship avionics will mitigate against switches that are stuck closed.

3.11 Computational nodes aboard a starship will be connected by three-dimensional technology that mitigates, in order of priority: a) short circuits, b) mass, and c) signal delay.

Let us unfold the case for 3.9, 3.10, and 3.11. Under worst-case or probabilistic fault models, the likelihood that healthy nodes will form a quorum is maximized by excluding faulty processors from the diagnosis or configuration algorithm [Somani and Agarwal 1992], [LaForge et al 1993], [Das et al 1993]. Further, given that each healthy node has knowledge of faults among the candidate nodes to which it might connect, a simple, distributed, greedy algorithm ("connect to only healthy nodes") suffices to form a quorum, as long as a quorum is feasible [LaForge and Korver 2000 MTAD]. In light of 3.8, the essential conditions for self-organizing quorums reduce to 3.7a) ensuring sufficient connectivity in a *point-to-point* architecture and 3.7b) having switching and interconnect that faithfully implements this architecture. That is, 3.9 is not only necessary, but, to the extent that 3.7a and 3.7b are satisfied, it is sufficient. Subrequirement 3.7a is central to our in-depth research, and is explained in Section 4. Analytic results reveal that switches stuck closed (but not switches stuck open) increase the redundancy (as well as the length of the longest wire) by *orders of magnitude* [LaForge 1999 Trans. Computers]. Further, requirement 3.9 draws a node's switches into its own fault containment region, in which case a single switch stuck closed threatens to inhibit exclusion of a faulty node from the quorum, or to reduce the quorum count by at least one (otherwise) healthy node. The ensemble of these observations provides motivation for requirement 3.10: we can tolerate switches that are stuck closed, but are well-served by technologies that hedge against such faults.

As Figure 10 suggests, requirement 3.11a follows by reasoning similar to that for 3.10. Moreover, and depending on our survivability confidence, the number of neighbors of each node will grow with the nominal number n of nodes. The attendant cost of layout (mass and wirelength) *dominates* the performance of chip and board-level electronics [Pedder 1993], [Lee and Cong 1997], [Geppert 1998]. However, *three-dimensional* processes, especially free-space interconnects ([Carson 1996], [Ishikawa and McArdle 1998], [Guilfoyle et al 1998]), ameliorate the tyranny of planar layout. For example, our best planar layouts for n-node binary hypercubes yield $O(n^2)$ area and $O(n)$ wirelength,[21] while locally spared hypercubes have area $O(n^2 \log n)$ and wirelength $O(n \log n)$.[22] In a metal-oxide-semiconductor (MOS) process, signal delay increases as the square of the longest wire. Hence the delay of the best planar layout for binary hypercubes scales as $O(n^4)$ in the absence of faults, $O(n^4 \log^2 n)$ for hypercubes that are locally spared. By contrast, a free-space multicomputer, such as that depicted in Figure 1, reduces to $O(n)$ the hypercube layout area ($O(n \log n)$ with locally spared redundancy), and diminishes internodal signal delay to negligible levels.

21. $O(g(n))$ denotes the *set* of real-valued functions no greater than $c \cdot g(n)$, for real $n > k$ and constants c, k.
22. Constructive upper bounds make use of the Strong Separator Theorem ([Ullman 1984] pp. 98-100), and are explicated in [LaForge 1994]. Unlike the case with, for example, binary (H-)trees, we do not have a matching lower bound, and so do not know how close to optimal are our upper bounds for hypercube layout and wirelength.



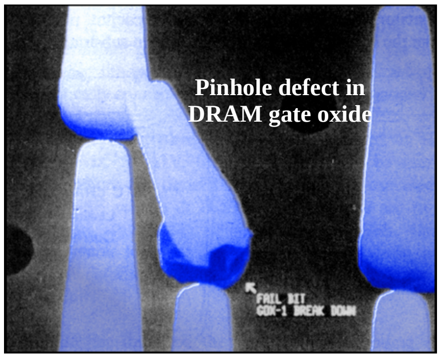


Figure 10: Tolerance to physical defects will remain a challenge to starship avionics.

We conclude our focus on circuit technology by approximating conditions for radiation survivability:

3.12 The combination of starship skin, shield, and avionics radiation hardening will be capable of withstanding cumulative exposures of 10^8 to 10^{10} rad(Si), as measured from the exterior of the skin.[23]

Dr. Steven Howe of the Los Alamos National Laboratory estimates that a starship will be exposed to radiation similar to that encountered in the solar system [Nordley 1998]. That said, the intensity and taxonomy of radiation varies considerably throughout the solar system. For example, vehicles in earth orbits below 10^4 kilometers are subjected to substantial numbers of protons, as well as the electrons that dominate higher orbits and escape trajectories. Shielding is much less effective against these high energy (up to 500 MeV) protons than with the relatively low energy (up to 7 MeV) electrons.[24] Further, when a vehicle boosts from low level to medium level orbits (say, from 800 kilometers to 1100 kilometers in a circular trajectory inclined 90° to the equator), the radiation intensity increases by threefold. Even with substantial shielding, the *Europa Orbiter* space probe avionics must tolerate Jovian doses of 1 Mrad(Si) [Guiar 1998].

Above the earth's radiation belts, typical background rates are on the order of 1 rad(Si)/second [Bendetto 1998]. To obtain 3.12 we have multiplied the latter value by the range of mission durations (15 to 1000 years, Table 2). However, this estimate does not take into account Linear Energy Transfers (LETs) that induce latchup and upset, nor is it fully cognizant of high density fluxes, such as those likely to be encountered when exploring near other planets [Lockheed Martin 1999]. Moreover, a substantial amount of work remains to gauge tradeoffs among shielding, process-level hardening, and architectural fault tolerance, where for that latter we are more interested in bit error rates than in radiation dosage. As mentioned in our Phase I proposal ([LaForge 1999 NIAC Phase I Proposal] item viii, p. 7), such a tradeoff study falls within the scope of Phase II. Section 5 includes this task in the context of enhancements to STAArchitecture.

Requirements 3.6 through 3.12 devolve from a perspective based on a spectrum of models, theories, analytic results, and technology. To successfully realize self-healing architectures and algorithms within the next ten to forty years, we must embody this spectrum in the form of dependable, convenient software for computer aided design (CAD). In addition, we must put into place new engineering, manufacturing, and project management practices and processes that make best use of CAD tools, as well as the knowledge these tools embody. The remainder of this section addresses these issues and how they interrelate.

Recalling the immune system paradigm of Avizienis and Hecht, low-level fault tolerance for starship avionics and software is analogous to our own cellular-level defense mechanisms. What would (and does) happen when these low-level mechanisms fail us? Although our higher level functions may remain capable

23. A measure of the radiant energy absorbed by semiconductors, 1 rad(Si) = 10^{-2} joules/kilogram [Bendetto 1998].
24. MeV = one million electron volts = 1.6×10^{-13} joules ([Kaufmann and Freedman 1999] p 122).

of defending us against macro-level threats (such as rush-hour drivers), we run the risk of succumbing to a common cold. Under conditions other than those for which they were originally designed, electronics and software components rarely exhibit effective defenses [Zorian et al 1999], [Lowry 1998]. Unbridled reuse of such components risks our system succumbing to the bit-level analogy of a common cold. This pertains in particular to commercial, off-the-shelf components (COTS):

3.13 Unbridled reuse of software or hardware components, with or without wrappers, impedes self-healing starship autonomy. This applies to COTS in particular.

[Avizienis 1997] articulates the case for considered, conservative application of COTS, a position amplified by York University Professor John McDermid [McDermid and Talbert 1998]:

> … It would worry me if people in aerospace and nuclear power industries started using a COTS solution without a clear demonstration that it fits all requirements … A major drawback to wrappers … is that you may need to make them extremely complex to interact with the COTS component.

On the other hand, a component need not be of strictly commercial origin in order to be reused – or *mis*used. Perhaps the most dramatic misuse of previously constructed components is the explosive failure of *Ariane 5*. The fault that led to this failure was a consequence of plugging *Ariane 4* flight software into successor spacecraft, without taking full account of the behavior of the reused component *viz*. variations in launch sequence [Aviation Week 1997], [Jezequel and Meyer 1997]. Moreover, such malpractice is not the exclusive province of the European Space Agency. We underscore this point with two of our own NASA spacecraft design experiences – one software, one hardware.

Deep Space 1 inherited much of its software from *Mars Pathfinder*. Unlike *Pathfinder*, however, *DS1* carried no planetary rover. Recognizing the *Ariane* lesson, the *DS1* flight software team attempted to excise *Pathfinder* code pertaining to the *Sojourner* rover. However, removing the code rendered the *DS1* flight software incapable of being compiled and linked. *DS1* and *Pathfinder* team members expended at least three calendar weeks rectifying this problem. During that time, concurrent development on flight software was substantially impeded [Eldred 1997], [Dornheim 1998]. This example exemplifies the cost of reuse: usually hidden, frequently exorbitant.

As to hardware, the original X2000 design prescribed 1394 Firewire Bus controllers based on COTS intellectual property ("IP", expressed in Verilog or VHDL design languages). Firewire is not designed to tolerate faulty nodes, and the boot sequence forbids cycles in the point-to-point connectivity. Project requirements necessitated redundant wiring paths [Guiar 1998], thus introducing cycles. In consequence, the 1394 IP was subjected to conditions other than those for which it was originally designed. Proper accommodation of these conditions warranted modifying and testing the IP hardware. Instead, these modifications were relegated to software. Although feasible to some extent, this approach exposed the avionics to a number of debilitating low-level hardware faults [LaForge 1999 JPL D-16485].[25] This example illustrates a common reluctance to "pry open the black box"; when such reluctance prevails, "ticking box" is more apt. How we come to so readily accept such ticking boxes brings to light to a broader issue:

3.14 Starship design teams must embrace a new generation of CAD tools and attendant processes.

Reflecting on our examples with DiSTARS and X2000 Firewire, we should expect sub-optimal, shoot-from-the-hip avionics and software whenever engineers are forced to produce designs outside of their respective specialties. To some extent this a downside of contemporary emphasis on cheaper spacecraft [Woerner and Lehman 1995], [Broad 1999]. However, the point of the examples extends to all areas

25. Among other advances initially planned, 1394 Firewire was eventually descoped from X2000 [Chau 1999].[37]

of software/hardware co-design. The cost of applying expert knowledge can be managed, and even reduced, by employing processes that make judicious use of Computer Aided Design (CAD) [Bryant et al 1989]. Epitomizing this approach, the semiconductor industry has very effectively ameliorated the need for large numbers of specialists by using CAD tools that, conveniently and dependably, incorporate the rigor and knowledge of experts. In the domain of spacecraft avionics and software, the outlook for requirement 3.14 appears promising. Spearheading research in the integration of CAD tools and processes, for example, NASA's Langley Research Center has partnered with the University of Virginia in creating the Intelligent Synthesis Environment (ISE) [Goldin et al 1998]. At the Jet Propulsion Laboratory, Mike Dickerson heads the recently-formed Design Center, whose charter mirrors that of ISE [Chau 1999].

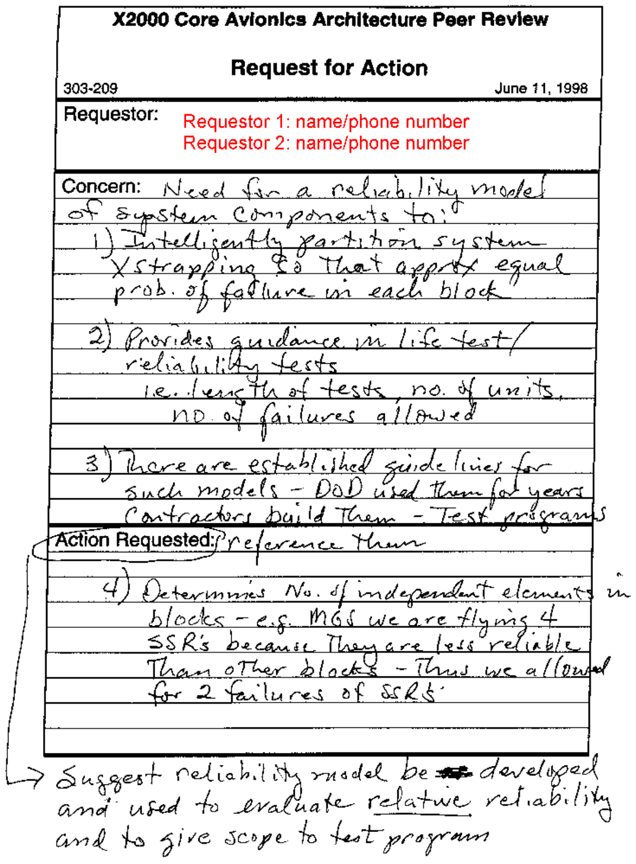
The promising outlook for 3.14 is bolstered by emerging levels of accessibility and capability for individual CAD tools. For example, automated theorem proving can be used to rigorously establish the correctness of software. Once thought to be an academic exercise, this technique is now a practical reality [Neumann 1996]. By way of illustration, the Automated Software Engineering group at NASA's Ames Research Center has created Amphion, a tool that mechanizes system-level verification in the face of evolving changes in application software; Amphion has been applied to problems in fluid dynamics and space shuttle navigation [Lowry 1998], and was used to find flaws in the *DS1* Remote Agent not uncovered through testing [Feather 1999]. At the Jet Propulsion Laboratory, researchers have repaired *Cassini* flight software by employing SPIN, a verification tool akin to Amphion [Schneider 1999 E1].[26]

Recent advances in architectural CAD tools combine computer software with classical hardware domains, such as power engineering. At Penn State University, for example, Mary Jane Irwin and Vijaykrishnan Narayanan have developed a tool that enables the designer to predict electrical power consumption based on software instruction mix [Irwin and Narayanan 1999]. At the level of system reliability, designers are now able to conveniently manipulate graphical interfaces that detail the dynamics of architectures subjected to faults. This capability is demonstrated by tools such as Relex$_{®}$, developed by Relex Software Corporation [Relex 1998], and MEADEP, developed by SoHaR [Tang et al 1998]. With respect to architectural requirements 3.6, 3.7, and 3.8, our own STAArchitecture program allows the user to synthesize and analyze computational avionics. As described in Section 4, STAArchitecture maximizes the probability of diagnosing and configuring a healthy quorum, while simultaneously minimizing cost and latency.[27]

To be sure, there remain gaps between what CAD tools are capable of doing and what they should do. Dependability analyzers, such as Relex$_{®}$ and MEADEP, tend to be very good at revealing behavior of a given design ("what is the probability of 100 nodes surviving and computing together for 50 years?"), but fall short when it comes to computing optimal allocations of resources, such as redundancy ("what is the minimally redundant way of connecting nodes so that at least 100 survive and compute together for 50 years?") Refer to Figures 7 and 8. Even the simplest techniques for organizing redundancy, such as local sparing, seem to be have been overlooked. Tools such as STAArchitecture serve to bridge these gaps.

Much more serious than gaps in CAD tool functionality: organizations frequently *starve* the adoption of new tools and processes, through either shear inertia or by active resistance. Economies of scale may account for some starvation: by comparison with semiconductor products, spacecraft design is a relatively small niche. Nevertheless, we would be remiss to understate the importance and difficulty of integrating breakthrough CAD tools and processes into spacecraft development teams. Figure 11 illustrates this widespread problem, one which severely impedes the advent of self-healing avionics and software.

26. Amphion is based on process algebra; SPIN uses model checking [Schneider 1999 E2].
27. *Cf.* Figures 15 through 20. It is beyond our scope to attempt a comprehensive survey of CAD tools for software/hardware co-design; such a task is being undertaken by JPL's Design Center [Chau 1999]. Rather, our objective is to indicate the importance and benefits of architectural-level CAD to self-healing autonomous starships.



X2000 Core Avionics Architecture Peer Review

Request for Action

303-209 June 11, 1998

Requestor: Requestor 1: name/phone number
Requestor 2: name/phone number

Concern: Need for a reliability model of system components to:
1) Intelligently partition system Xstrapping so that approx equal prob. of failure in each block
2) Provides guidance in life test/ reliability tests i.e. length of tests, no. of units, no. of failures allowed
3) There are established guidelines for such models - DOD used them for years Contractors build them - Test programs

Action Requested: reference them

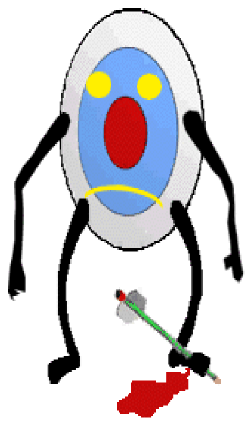
4) Determines No. of independent elements in blocks - e.g. MGS we are flying 4 SSR's because they are less reliable than other blocks - Thus we allowed for 2 failures of SSR's

→ Suggest reliability model be developed and used to evaluate relative reliability and to give scope to test program

Lack of receptiveness to rigorous design processes, in this case on the part of X2000, NASA's flagship project for multi-mission avionics.

Project lead's [a] verbal response to this request for action:

"We don't do probabilistic analysis. It is too complicated and, even if you did calculate the numbers, I wouldn't believe them. You look at the worst case, you build as many spares as you think you need, and you go for it."



… and afterwards:

No written response to the senior NASA engineer (Requestor 1) [a] who wrote this critique. Worst-case analysis and recommendations, though subsequently carried out, were ignored.

a. Both of these people are respected, talented professionals. The purpose of the example is not to inflame, but rather to encourage acceptance of new tools and processes.

Figure 11: By resisting new CAD tools and design processes, organizations shoot themselves in the foot. The problem impedes near-term missions as well as self-healing autonomous starships.

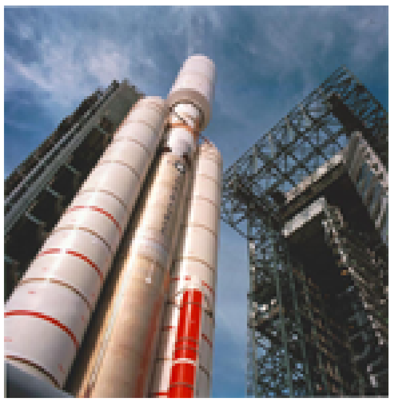
As Figure 11 suggests, a peculiar pattern of human resistance often arises when new CAD tools or design processes promote and incorporate rigorous models.[28] With the notable exception of *Cassini* [Marcus 1998], this has resulted in a conspicuous lack of attention to detail among recent missions. Maintaining an informal team of generalists was, in fact, a deliberate strategy for *Mars Pathfinder* [JPL 1994 Pathfinder Software], [Muirhead 1996 Pathfinder Design], [Muirhead 1996 Pathfinder Test]. The success of *Pathfinder* has led subsequent missions to adopt similar cost-cutting measures [Hotz 1999 Dec]; [29] *e.g.*, i) design documentation limited to viewgraphs and email; ii) decisions and rationale for decisions not systematically captured by meeting minutes.[30] Many of these *Pathfinder* practices (or omission of practices) run counter to NASA's Strategic Plan for risk management [Lalli 1998], or NASA reliability standard NSS 174013 [Voas et al 1997]. Abandoning rigor and attention to detail is exactly the opposite of what we need to do in order to construct self-healing software and avionics for starships [Hecht and Fiorentino 1988].

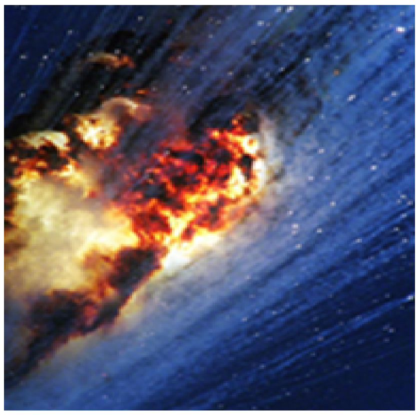
28. Another example of such resistance is provided by a reviewer of a proposal to NASA's Small Business Innovative Research (SBIR) Program: "…limited utility. Analysis of graph theoretic connectivity is largely not a problem in the design of most fault-tolerant systems." Perhaps not, but such analysis should be part of the design of fault tolerant systems! Graph theoretic fault tolerance is a consequence of requirements 3.6 through 3.8, and is the focus of Section 4. Had graph theoretic fault tolerance been properly applied, X2000 architects would have used 18 serial bus wires per node instead of 36 [LaForge and Korver 2000 Graph Fault Tolerance]. Ironically, the SBIR topic called for "mathematics-based methods for specification, design, and analysis of digital systems" [NASA 1999].
29. *Mars Climate Orbiter*, *Polar Lander*, and *Deep Space 2* were not successful [Hotz 1999 Nov], [Wilford 1999].
30. Compare Table 5 of [LaForge 1999 JPL D-16485] with findings of the *Mars Polar Lander* investigation: "faulty communications" between spacecraft designers and assembly team, technical dialogs "inadequate" [Broad 1999].

Unless and until design teams embrace the new generation of CAD tools and processes, we remain deprived of the full benefit of software such as MEADEP, Relex$_{®}$, SPIN, Amphion, and STAArchitecture. Conversely, embracing such tools, and the processes they embody, will enable spacecraft that are more dependable and more capable. This pertains especially in the case of self-healing starship autonomy.

Having examined requirements for starship software and avionics from the standpoint of computer aided design, let us consider three key issues rooted chiefly in *process*. We begin by observing that traditional design of software and avionics for long duration missions places heavy emphasis on operational reliability, and this is proper. What is not proper, however, is the extent to which we tend to underemphasize the need for making the software and avionics work in the first place.[31] Frequently, and mistakenly, we downplay the importance of manufacturable *yield*. As is the case with CAD tools, spacecraft design teams would do well to draw on a lesson long since learned by the semiconductor industry:[32]

3.15 Designing starship reliability without regard for yield is like borrowing money at loanshark interest: to realistically satisfy budgets and schedules requires high levels of reliability *and* manufacturing yield.


Air Force Titan 4A
prior to 12-Aug-1998 launch


Titan 4A explodes 41 seconds after launch,
a $1B loss second only to that of *Challenger*

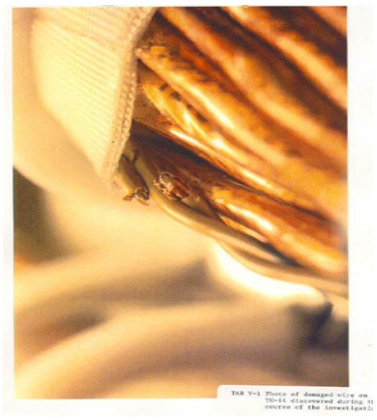

Faulty wiring that
led to Titan 4A failure

Figure 12: Manufacturers had a difficult time producing working copies of the wiring harness that caused this operational failure [Halvorson 1999]. The explosion dramatizes how low yield leads to low reliability. Conversely, systems with high yield tend to be more reliable. (U.S. Air Force photos)

As Figure 12 depicts, it is important to consider yield and reliability holistically, not only at the level of basic components, but at all levels of integration. In the vein of *modular* manufacturability (*cf.* page 6), Ted Marcopolus of Hewlett-Packard Corporation has demonstrated how to economically achieve yield and reliability, largely through measurements carried out with his *Lunar Prospector* Electrical Test-Set (LETS) [Marcopulos 1998].[33] Similar, untapped opportunities await us in the domain of *software*:

3.16 The teams that build onboard or earth-based software for starships will instrument failures and faults of that software, at quantifiable levels commensurate with how we measure physical aspects of avionics.

31. By "work in the first place" we mean that each subsystem passes test at all levels of integration.
32. The semiconductor industry in fact accords yield higher priority than reliability. There is also an unfortunate lack of communication between yield engineers and reliability engineers. We do not advocate this latter practice for spacecraft design teams! Rather, we recommend adopting an economics-based view of dependable systems, with yield and reliability two sides of the same currency. This position is elaborated by [Lalli 1998].
33. Interestingly, *Lunar Prospector* was controlled from earth, and did not have an onboard flight computer.

As with hardware, software systems tend to be more reliable if they begin their operational life with fewer faults ([Koshgoftaar et al 1998] p. 71). *Avoidance* is, and will remain, the most economical approach to handling software faults ([Siewiorek and Swarz 1982], Chap. 3).[34] Furthermore, a great deal – perhaps most – of the complexity of a starship will be encompassed by its onboard and earth-based software. For *electronics* we measure, in excruciating detail, quantities such as current leakage, doping concentrations, and variations in oxide thickness [Bendetto 1998], [Runyon 1999], [Taur 1999], [Zorian 1999]. These quantities guide changes to our development processes – changes which converge on hardware that is suitable for operation. In the case of *software*, however, we have yet to deploy instrumentation that quantifies the relation between failures and faults, faults and their root causes [Glass 1997].[35] The viability of starship software depends on possessing, and taking advantage of, such instrumentation [Meyer 1999].

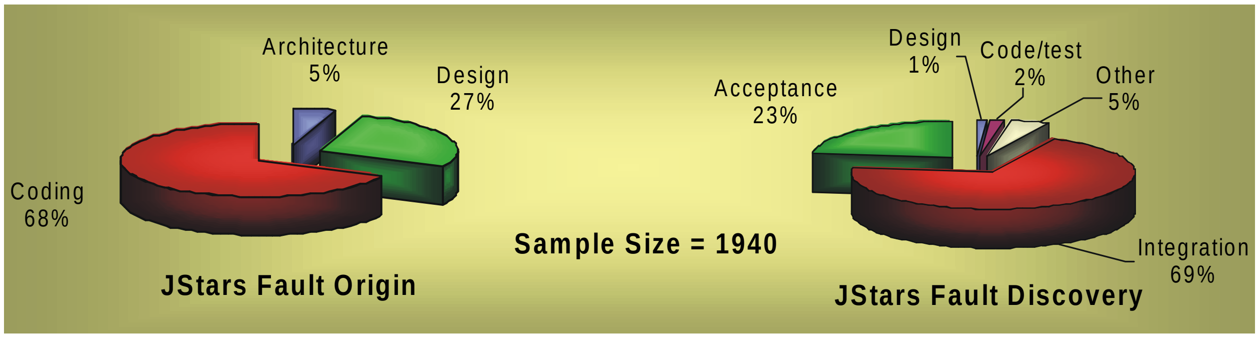


Figure 13: The best instrumented software yet reported illuminates overall trends [Koshgoftaar et al 1998], but falls short of pinpointing relations between failures and faults. It also remains to quantify the rate of fault discovery against application and test code intensity, in the domain of spacecraft. Attempts to gather such statistics with *Deep Space 1* and *Cassini* were unsuccessful, largely due to lack of programmatic support ([Nikora et al 1998], [LaForge 1997 CVS], [LaForge 1997 DS1]) and tools (*cf.* discussion of 3.14).

For illustration, suppose that we observe 100 mission-critical failures in the software module responsible for sending science data to earth. Further suppose that, over the same development cycle, we observe 10 mission-critical failures in the software module that manages the distribution of spacecraft power. At first blush, the order of magnitude difference in module failure rates suggests that we better spend more project dollars testing communication software. But wait! This line of reasoning is based on the tacit assumption that the two modules are of comparable complexity, and that they have been subjected to testing at approximately the same level of intensity. If these assumptions do not hold, then uncovering, say, 95% of mission-critical faults in communication software could cost much less than uncovering 95% of mission-critical faults in the power distribution software. In this case, we would do better to spend more project dollars testing power distribution software. To effect a decision that accounts for all of these factors, we need a predictive model that tells us how to minimize the test intensity (cost), while maintaining, say, 95% fault coverage (benefit), as a function of application complexity (independent variable). Alternatively, such a relation can help us predict the fault coverage, as a function of application complexity and test intensity. The benefit of such a predictive model is born out in the avoidance of software design faults.

To develop such a predictive model, we need to instrument the software development process in order to 1) trace failures from the point at which they are observed to the point at which the underlying fault(s) were inserted into the system; 2) measure the evolving complexity of the application software; 3) quantify

34. It is in general much more expensive to tolerate latent software faults that manifest *in situ*.
35. An unmanned NASA mission may have as many as three separate problem reporting and tracking systems; *e.g.*, software only (PFR), mission level (PR), and "institutional" (FR). As intimated by the caption to Figure 13, such systems do not at present satisfy requirements for quantitatively instrumenting software failures and faults.

the rate, type, and severity of faults; and 4) quantify test intensity and the way it changes over time [Munson and Werries 1996]. Patterns between failures, faults, program structure, and test intensity will emerge, and the fidelity of our model will continually improve [Nikora et al 1998]. At a tactical level, such work will enable engineers and managers to answer questions such as: How many faults are there? In what modules are the faults? How quickly are they being removed? Where should fault removal efforts be applied, and to what extent? Refer to Figure 13. The Army/Air Force Joint Surveillance Target Attack Radar System (JStars) appears to be the only project for which a significant volume of such data, coarse-grained as it is, has been reported ([Koshgoftaar et al 1998], [Voas et al 1997] "data elusive"). Development of an instrumented software model, and its application to work we propose for Phase II, would synergistically benefit starship autonomy.

As the caption to Figure 13 alludes, programmatic support (or lack thereof) has a prevailing influence (positive or negative) on advances in development processes [Cole 1999]. The discussion accompanying requirement 3.14 underscores how this applies as well to CAD tools and to the modeling techniques CAD tools embody. *Mars Pathfinder* and the X2000 Mission Data System software exemplify how teamwork between engineers and managers enable successful implementation of architectures and algorithms [Woerner and Lehman 1995], [Muirhead 1996 Pathfinder Design], [Muirhead 1996 Pathfinder Test], [Rasmussen and Sacks 1998]. At the opposite end of the spectrum, marginalization of the *Deep Space 1* Remote Agent software demonstrates how self-healing autonomy can suffer in the face of programmatic obstacles [Savino 1997].[36] Figure 14 recounts an analogous setback in the domain of hardware.[37] In summary, the last item in our list of enabling characteristics may be the most important of all:

3.17 On par with any technical consideration, administrators and managers of resources must actively promote, and participate in the development of, self-healing starship autonomy.

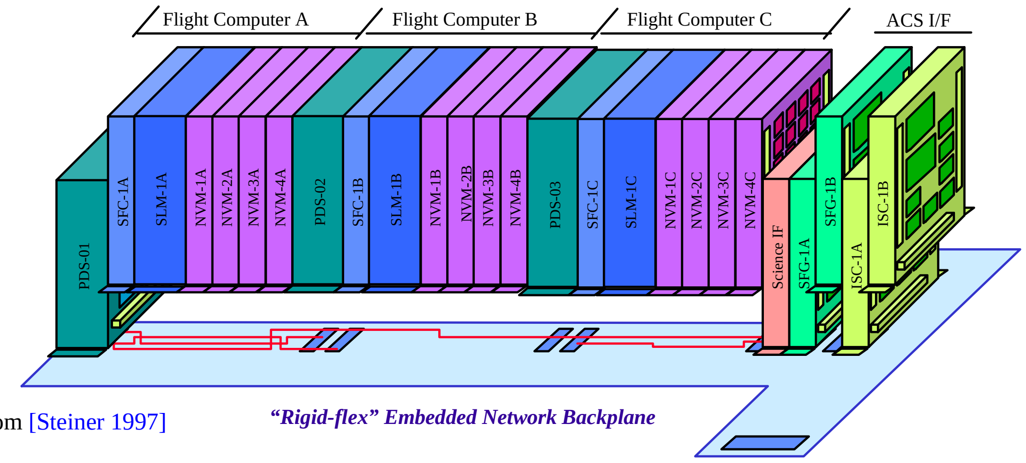


Diagram from [Steiner 1997]

Figure 14: Wafer stack adopted by X2000, then cancelled.[37] Retreats of this nature snuff out breakthroughs in self-healing autonomy. Conversely, programmatic support enables self-healing autonomy.

36. "RA" became "RAX": initially planned for controlling the entire *DS1* spacecraft, the Remote Agent was relegated to experimental status; RAX successfully carried out three out of four scheduled tests over a four day period of *in vitro* activation [JPL DS1 Press Release], [Gluck 1999], [Bernard 1999].

37. Since the 20-Aug-1998 Preliminary Design Review [Chau 1998 PDR], X2000 has scuttled: i) the multi-chip module (MCM slice) backplane and packaging [Hunter 1997], [Hunter 1998]; ii) allocation of one computer or microcontroller per slice [Chau 1998]; iii) the MCM slices themselves [Steiner 1998], [Chau and Holmberg 1998] (reverted to printed circuit boards); iv) high bandwidth 1394 Firewire Bus for internode communication (Firewire controllers remain, but, in the absence of microcontrollers, are without purpose) [Chau 1999]. In fact, X2000 marks a second retreat from stacked wafer avionics, previously planned for *DS1* [Alkalai and Geer 1996], [Savino 1997].

Themes	Points of Interest	Details
Modeling	Akin to the use of imaginary numbers to characterize alternating currents, the game of Connect-the-Dots serves as a fountainhead for self-healing architectures.	Sec. 4.1 p. 22
Distance within architectures	Connect-the-Dots based on metric spaces defined by structure and labeling. Latency modeled by radius and diameter.	Sec. 4.2–4.5 Tables 7–15
Performable diagnosis and configuration	Global properties of self-healing systems emerge as a consequence of locally-specifiable properties of connectivity.	Sec. 4.2– 4.5
CAD tools	STAArchitecture optimizes interconnection, maximizes design leverage.	Figs 15-20
Key to performability	What $(f+1)$-connected n-node graphs have a) fewest edges and b) minimize the maximum radius or diameter of quorums induced by deleting up to f of the n nodes?	Sec. 4.1.1
Performability	Clique-based K-cubes preferred to cycle-based C-cubes	Fig. 21
Minimum size, f-fault-tolerant	Worst-case: quorums guaranteed by chordal graphs of [Hayes 1976]: poor latency, diagnosable at size quadratic in a constant proportion np of faulty nodes.	p. 27 Sec. 4.6
Trees versus connected components	Trees overconstrain what we need to build. Insisting on configuration of a tree discards edges that could be used to carry signals, artificially limits throughput, leads to unnecessary re-initialization. Target quorum should not be limited to a tree that spans healthy nodes, but rather should include all edges between healthy nodes.	Sec. 4.2.1 Fig. 22–24
Algorithms for diagnosis, configuration	Maximum parallelization (minimum serialization) achieved with connected component heuristic $\mathcal{A}_{\text{connected}}$. Under blocking constraints, efficiency bounded by the number of matchings in a one-factorization,[63] and by the arboricity.[55]	Fig. 22–24 Fig. 30–32
Notation	Index of terms, definitions, symbols	Table 6
Lower bound on radius	Fault-tolerant variations on a bound attributed to Moore: inequalities (1) and (2)	Table 7 Sec. 4.5
C-cubes dimension d radix j	Suboptimal: diverge from the Moore Bound. Distance governed by modulo-j variation on L_1 metric. Lower bound on quorum radius and diameter gleaned by explicating surface area and volume of balls in this metric; upper bounds by construction.	Sec. 4.2, 4.3 Sec. 4.5
K-cubes dimension d radix j	Optimal: in a ratioed asymptotic sense, converge to Moore Bound for bounded d. Distance governed by Hamming metric. Intersects with C-cubes when radix is less than 5. Distinct from C-cubes for radices greater than or equal to 5.	Table 7 Sec. 4.2, 4.4 Table 11
K-cube-connected cycles, edges	Optimal: in a ratioed asymptotic sense, converge to Moore Bound when dimension, cycle length sufficiently small. Distance governed by a hybrid Hamming-cyclic metric. Fill in gaps between points of constructibility for K-cubes.	Sec. 4.4 Tables 7, 12 Tables 13, 14
Stars, cycles, cliques	Unique graph architectures with minimum size, minimum quorum radius and diameter, at minimum ($f = 0, 1$) and maximum ($f = n$-2, n-1) fault tolerance.	Table 7 Table 15
Probabilistic diagnosis, configuration	STAArchitecture delivers on requirements 3.6 through 3.8. Tolerance to constant proportion of faults *is* feasible: quorums self-diagnose, self-configure. Size of redundant architecture: $\Theta(n^2)$ to $\Theta(n \log n)$ to $\Theta(n)$, depending on stringency of conditions	Sec. 4.6 Table 16 Fig. 30–32
Unfinished business	Characterize architectures satisfying 4.1.1, with $f = np$, in the probabilistic and worst cases. Rigorously characterize MTAD with imperfect test coverage.	4.6.2, 4.6.3 4.6.4

Table 5: Technical highlights of Section 4. Much of this material will be published in [LaForge and Korver 2000 Graph Fault Tolerance] and in [LaForge and Korver 2000 MTAD].

4. Advances in Self-Healing Architectures and Algorithms

In broad strokes, Sections 2 and 3 paint a vignette of *challenges* and *opportunities* for co-design of starship software and hardware. Refer to Tables 3 and 5. Wielding a somewhat finer brush, this section portrays our in-depth progress on *solutions* that address requirements 3.6 through 3.9, as well as 3.14. We introduce these solutions by way of an historical analogy and a practical example.

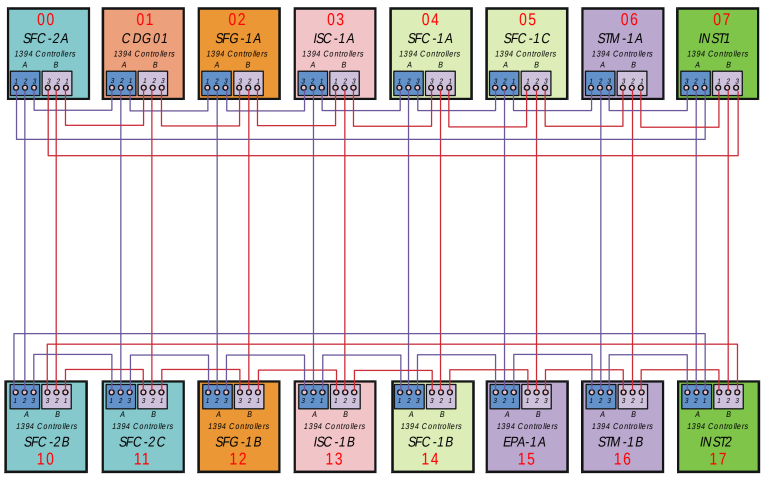
4.1 Configuration: STAArchitecture Connects the Dots

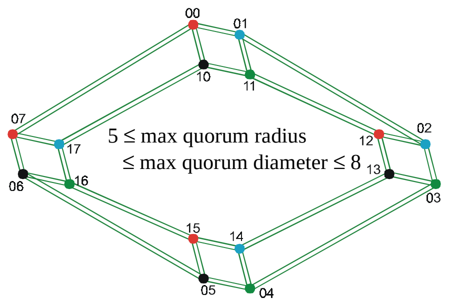
Charles Proteus Steinmetz is perhaps best known for showing us how to use imaginary numbers to predict what electricity will do. Indeed, Cornell Professor Vladimir Karapetoff observed that Steinmetz, when Chief Consulting Engineer for the General Electric Company, was "allowed to try to generate electricity out of the square root of negative one". Adds the *New York Times*:

That, doubtless, was what the man [Steinmetz] often seemed to be doing to those to whom mathematics as he knew it was equally incomprehensible and useless. Fortunately his employers – no genius ever had better and few as good – took a different view. ([Elfun 1977], p. 69)

While we do not purport to be of the same caliber as Steinmetz, our fundamental approach is similar to his. Instead of generating electricity out of the square root of negative one, we synthesize self-healing architectures and algorithms by playing Connect-the-Dots [Knowledge Adventure 1999].

For illustration, suppose that you are designing computational avionics whose n processing nodes (the dots) are connected by point-to-point *edges*: each edge carries signals over one or more parallel channels.[38] Further suppose: i) the manufacturing and operational cost of your system is dominated by *degree* – that is, by the number of edges (count of channels or pins) emanating from each node; ii) the latency of a signal is dominated by *pathlength* – that is, by (one plus) the number of nodes that the signal must traverse. Your primary design objective (4.1.1a) is to maintain a quorum in the presence of a bounded number f of *partitioning faults* – that is, failed nodes that block signals.[39] Since faults may be distributed in an arbitrary fashion, f is the *worst-case* fault tolerance. Figure 15 illustrates for $(n, f) = (16, 2)$.





CDG	Command Data Ground support equipment
EPA	OpComm Electronics & Processor Assembly
INST	Instrument Interface
ISC	IMU/Sun Sensor Controller
SFC	System Flight Computer
SFG	Stellar Frame Grabber

Figure 15: 2-fault-tolerant graph architecture handcrafted by two experienced designers, node degree = 6.

38. *Cf.* 3.7a and b, p. 13. Example based on initial X2000 avionics design depicted in Figure 14. Also see [LaForge 1999 JPL D-16485], [LaForge and Korver 2000 Graph Fault Tolerance], and references cited in footnote 37.

39. Partitioning faults are not necessarily nodes that have *failed silent*, but for the purpose of our illustration the two may be considered equivalent. Because vertex connectivity is no greater than edge connectivity,[43] bounding the number of partitioning faults conservatively encompasses failures in the edges.

In the interest of performance, your secondary objective (4.1.1b) is to minimize communication *latency* [Agarwal 1991]. Each processing element of our architecture corresponds to one of *n* nodes in a *graph*, while the edges of the graph prescribe a "directly connected to" relation between pairs of nodes.[40] The number of edges in a path is its *length*.[41] The (graph) *distance* between two nodes is the length of the shortest path that connects them (or infinity, if there is no such path). The *eccentricity* of a node is the maximum distance to another node in the graph. The graph *radius* is the minimum eccentricity, taken over all nodes. The graph *diameter* is its maximum eccentricity.[42] A graph is *connected* if is has finite radius or diameter; equivalently, if there is a path between every pair of nodes. The (vertex) *connectivity* of a graph *G* is the minimum number of nodes whose removal from *G* results in a disconnected graph, or a lone node.[43] Since a partitioning fault is equivalent to removing a node from a graph, we are especially interested in subgraphs *induced* by removing as many as *f* nodes, along with all edges belonging to the removed nodes. An induced subgraph that consists of a single connected component is a *quorum*. Radius and diameter are natural measures of latency, and this leads to our fundamental question about performable structures:

4.1.1 What (f+1)-connected *n*-node graphs have a) fewest edges and b) minimize the maximum radius or diameter of quorums induced by deleting up to *f* of the *n* nodes?

Starship architects need help answering 4.1.1, even with *n* as few as 16. The engineers who collaborated for *weeks* on the design of Figure 15 believed (correctly, but without the benefit of rigor) that their solution would tolerate two faults.[17] Especially striking is the *doubling* of edges, intention of which was to tolerate one, perhaps three broken wires (our engineers weren't sure), in addition to two faulty nodes. Such lack of crisp formulation tempts *requirements drift*: how many faults to tolerate … two? five?[44] As Figures 16 through 20 demonstrate, the proper tool, incorporating knowledge of graph architectures, can clarify such questions, provide optimum or near optimum solutions, and minimize requirements drift. Over a range of benefits (fault tolerance, latency) and cost (wires or pins per node), our STAArchitecture software, devel-

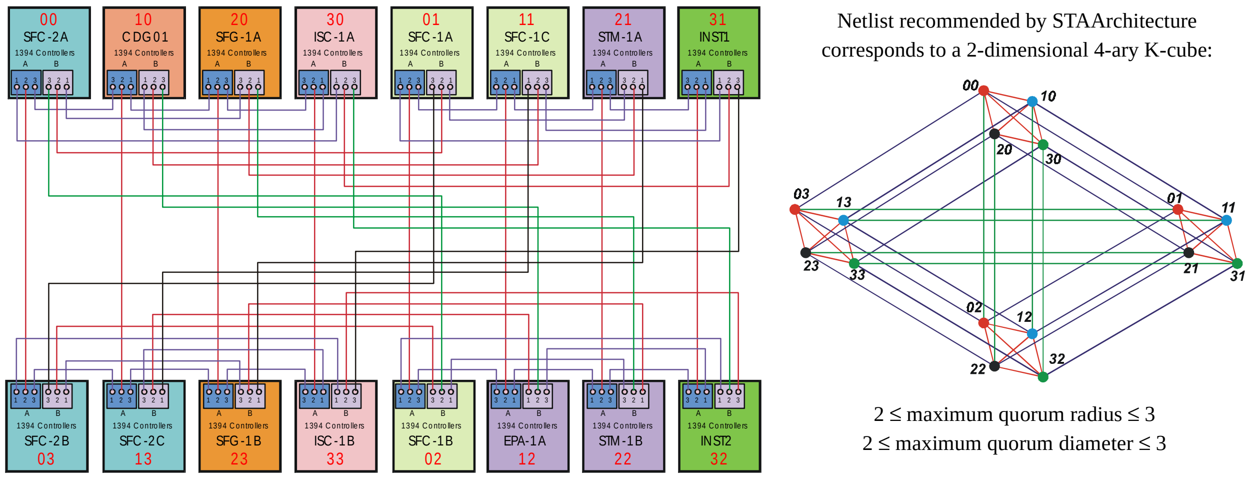


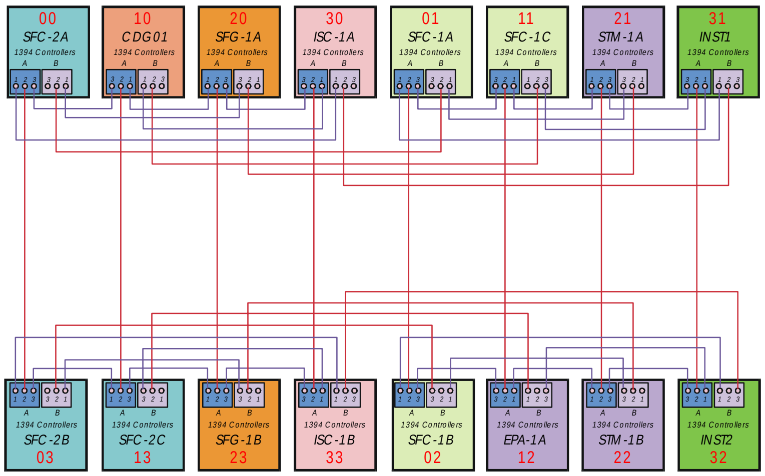
Figure 16: Same cost (node degree = 6) as design of Figure 15, but improved fault tolerance (5) and latency.

40. If more than one edge may connect nodes then we have a *multigraph*, example of which is depicted in Figure 15.
41. In a path, each edge may be traversed only once. Multiple traversals of some edge give rise to a *walk*.
42. The diameter is at least the radius and at most twice the radius ([Chartrand and Lesniak 1986], Thm 2.4).
43. "Node" and "vertex" are interchangeable. Vertex connectivity is to be distinguished from edge connectivity; the latter equals the minimum number of edges whose removal results in a disconnected graph or a lone vertex:

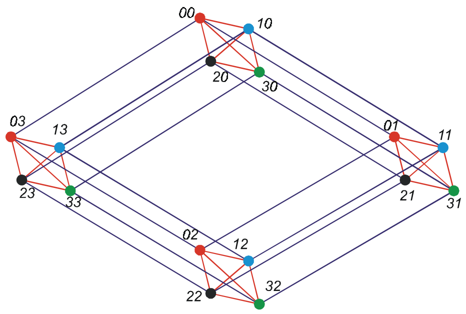
vertex connectivity ≤ edge connectivity ≤ minimum degree ([Chartrand and Lesniak 1986], Thm 5.1:

44. What about combinations of broken wires and nodes? The inequality stated in footnote 43 implies that the worst-case cost of tolerating a broken wire is at least as great as that for a faulty node. Furthermore, equality (and at minimum cost) is achievable for every *n* and *f* [Hayes 1976]. It therefore suffices to consider faulty nodes only.

oped as part of this Phase I effort, recommends architectures that are optimal or near-optimal. When engaged in this game of Connect-the-Dots, STAArchitecture outplays the human hand.



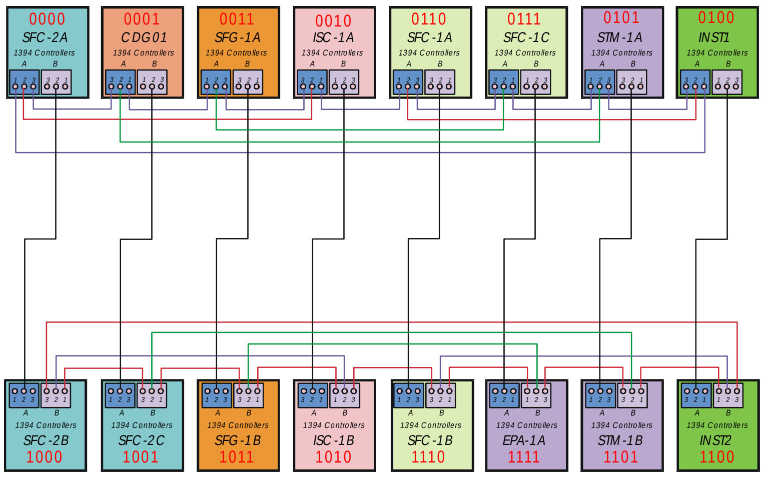
Netlist recommended by STAArchitecture corresponds to a 1-dimensional 4-ary K-cube-connected cycle, with four nodes in each of 4 cycles:



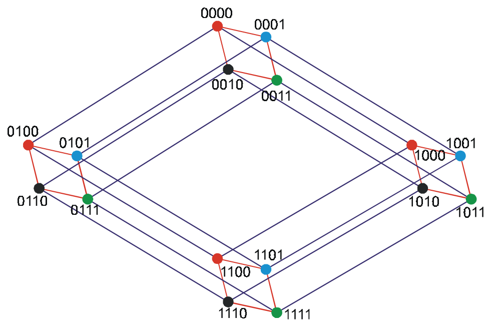
maximum quorum radius = 3

3 ≤ maximum quorum diameter ≤ 4

Figure 17: Less cost (node degree = 5) than with Figure 15, but improved fault tolerance (4) and latency.



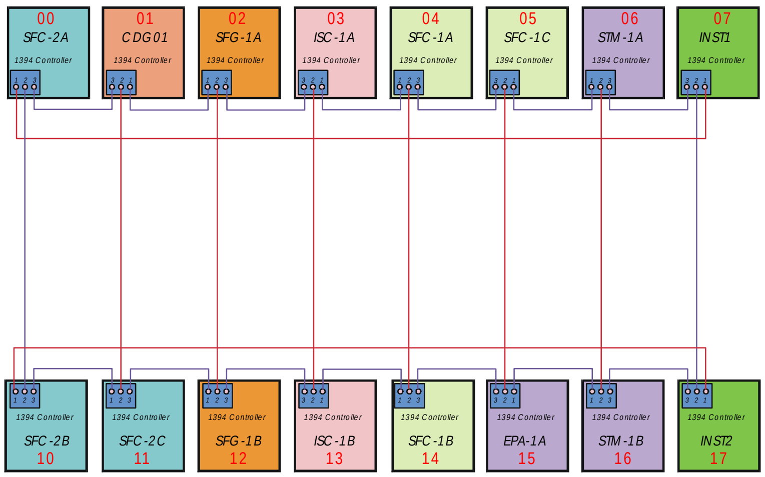
Netlist recommended by STAArchitecture corresponds to a 4-dimensional binary hypercube:



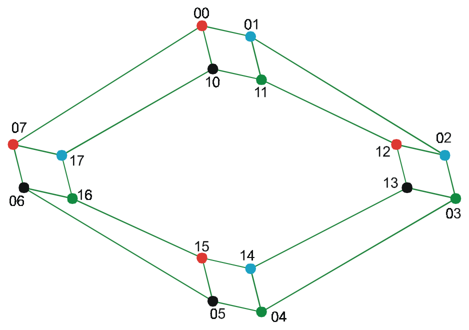
4 ≤ maximum quorum radius ≤ 5

4 ≤ maximum quorum diameter ≤ 5

Figure 18: 3-fault-tolerant choice, value again superior to that of Figure 15: node degree = 4, less latency.



Netlist recommended by STAArchitecture corresponds to a 1-dimensional binary K-cube-connected cycle, with eight nodes in each of two cycles:

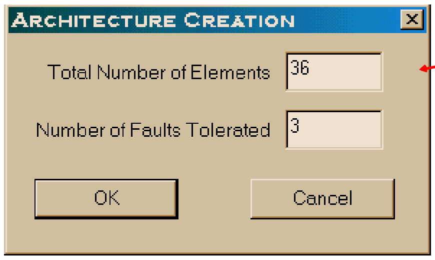


maximum quorum radius = 5

5 ≤ maximum quorum diameter ≤ 8

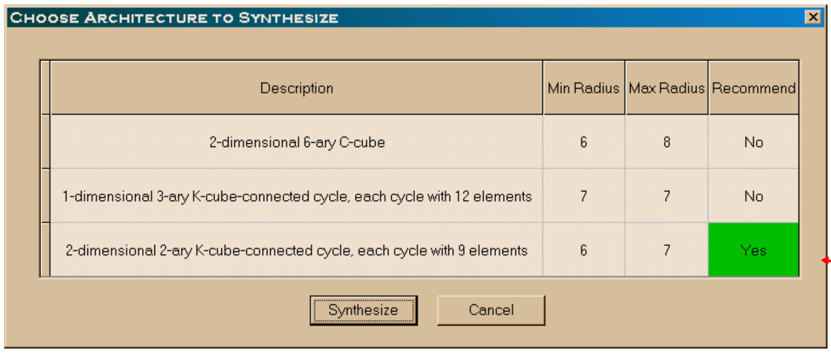
Figure 19: Same fault tolerance (2) and latency as in Figure 15, but at half the cost (node degree = 3).

For given node count and fault tolerance, STAArchitecture automatically generates architectures with minimum wire-count and minimum latency. Alternatively, you can use STAArchitecture to handcraft your own designs.



Total Number of Elements	36
Number of Faults Tolerated	3

① Specify the number of computational nodes, as well as the maximum number of faulty nodes.

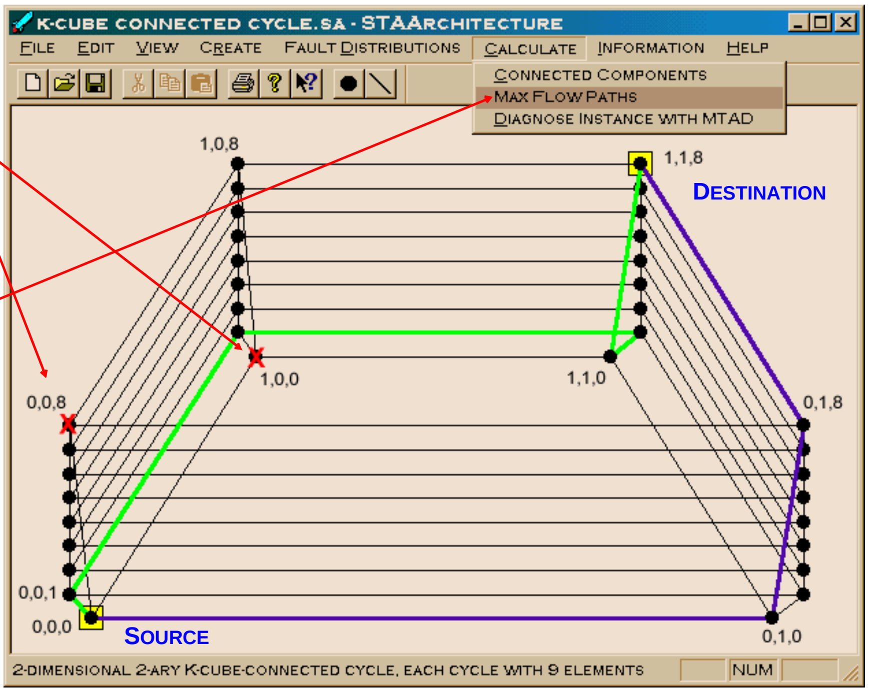


Description	Min Radius	Max Radius	Recommend
2-dimensional 6-ary C-cube	6	8	No
1-dimensional 3-ary K-cube-connected cycle, each cycle with 12 elements	7	7	No
2-dimensional 2-ary K-cube-connected cycle, each cycle with 9 elements	6	7	Yes

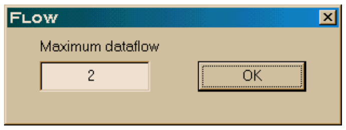
② STAArchitecture offers you design *choices* and *recommendations*.

③ Choose an architecture.
④ STAArchitecture synthesizes it for you.

⑤ Create an *instance* of the architecture by injecting **faults** into the architecture. The fault pattern may be generated by STAArchitecture, or you may craft the fault pattern by hand.

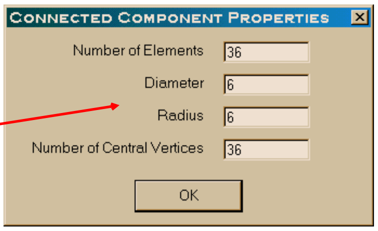


⑥ Review the *throughput* of the faulted instance using metrics such as *parallel dataflow*, calculated by STAArchitecture at your request.



Maximum dataflow	2

⑦ Check the *latency* of the faulted instance using metrics such as *radius* and *diameter*, calculated by STAArchitecture at your request.



Number of Elements	36
Diameter	6
Radius	6
Number of Central Vertices	36

The value of STAArchitecture increases with the number of nodes. STAArchitecture is especially useful for massively parallel systems (hundreds to thousands of nodes), where handcrafted schematic capture is unwieldy.

Figure 20: STAArchitecture synthesizes and analyzes architectures for self-healing multicomputers at a level where the designer has maximum leverage: interconnection of computational nodes.

4.2 Performability: STAArchitecture Applies Knowledge About Distance

Recalling the opening passages of Section 2, interstellar travel and self-healing architectures share a common theme: *distance*. By contrast to the L_2 distance used in celestial measurements,[45] we often employ the *graph distance* defined in Section 4.1.[46] We also exploit the L_1 (a.k.a. *Manhattan* or *city block*) distance defined by equation (6). In executable form, STAArchitecture brings to bear an arsenal of theorems and algorithms, many of our own creation. Let us explain how this arsenal is stockpiled.

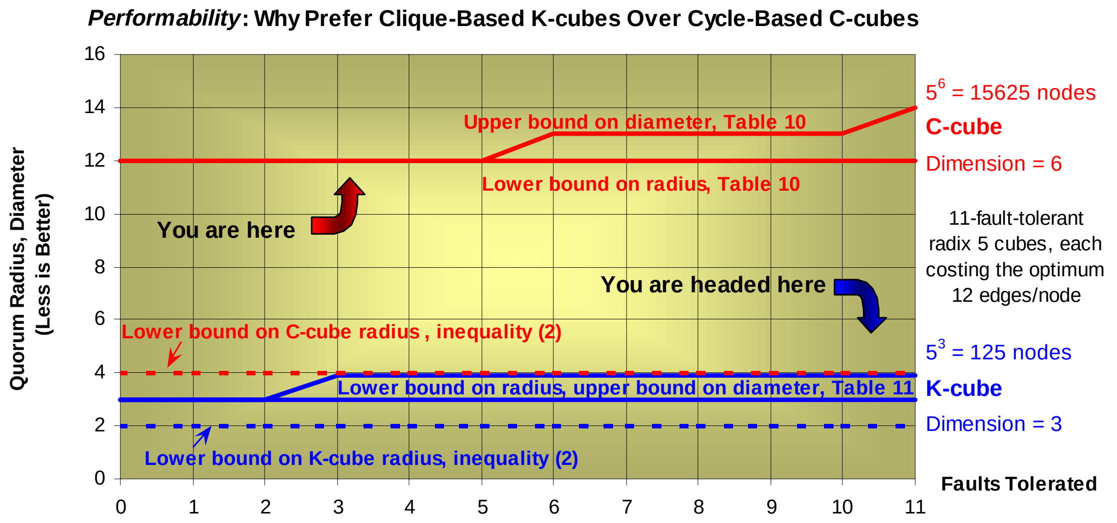


Figure 21: Performability measures the combination of fault tolerance and performance. The fractional fault tolerance of K-cubes (and their relatives, K-cube-connected edges and cycles) is superior to that of traditional C-cubes. Moreover, for given fault tolerance, and at minimum cost of channels and pins per node, the diameter of a K-cube is less than the radius of the corresponding C-cube. Furthermore, the radii of K-cubes approach the lower bound of inequality (1), whereas C-cube radii diverge from this bound.

In his seminal introduction of graph models for configuration,[47] [Hayes 1976] proposes and analyzes f-fault-tolerant architectures with minimum edge count, a.k.a. *size*.[48] A lower bound on size is readily seen by noting that the connectivity of a graph is at most the minimum degree of a node in the graph.[43] In consequence, the degree of every node in an $(f+1)$-connected graph (*i.e.*, f-fault-tolerant graph architecture) is at least $f+1$. If we sum the degrees of all the nodes then we have counted every edge twice. The size of any $(f+1)$-connected n-node graph is therefore at least $\lceil n(f+1)/2 \rceil$. For *any* integers $n > f > 0$, moreover, [Hayes 1976] constructively achieves this bound with *chordal graphs* of order n and size $\lceil n(f+1)/2 \rceil$ from which we can remove i vertices, $0 \le i \le f$, and still have an n-i vertices connected together as a path P_{n-i}.[49, 50] As long as quorum latency is not a significant concern, these chordal graphs effectively answer, and serve as a general solution to, 4.1.1a.

45. The *Euclidean* distance L_2 $(\mathbf{x}, \mathbf{y})$ is the square root of the sum of the squared differences of the respective coordinates of $\mathbf{x}$ and $\mathbf{y}$. Though standard, this notation conflicts with that used by mission planners for Lagrange points.
46. Following mathematical custom, we will use the "distance" and "metric" interchangeably.
47. The somewhat busy channel routing of Figure 16 serves as reminder to consider the area and wirelength of *layout*, as well as graph properties. A layout model (requirement 3.10, page 13) is particularly applicable to very large scale integrated (VLSI) circuits; for a detailed elaboration, refer to [Ullman 1984], [LaForge 1999 Trans. Computers], or [LaForge 1999 Trans. Reliability]. By contrast with VLSI layout, this section exposes properties of graphs in order to capture the cost of fault tolerant point-to-point structures such as buses or networks.
48. The *size* e and *order* n of a graph are the number of edges *resp.* number of vertices it contains.
49. For limited f, [Hayes 1976] also gives graph architectures for simple cycles and balanced trees.[50]

On the other hand, where latency is a concern (as in the case of high performance starship multicomputers), quorums configured from the chordal graphs of [Hayes 1976] are about as far from optimal as possible: i) the radius of $P_{n\text{-}i}$ equals $\lfloor (n\text{-}i\text{-}1)/2 \rfloor$, the maximum possible; ii) the diameter of $P_{n\text{-}i}$ equals n-i-1, again the maximum possible; iii) the radius of any quorum (not just a $P_{n\text{-}i}$) formed from a chordal graph is greater than that formed from a one-dimensional K-cube-connected cycle.[51] For illustration refer to Figure 21. At $(n, f) = (125, 11)$ or $(n, f) = (15625, 11)$ chordal graphs have radii 11 *resp.* 1302 ([LaForge 1999 JPL D-16485] Cor. 28.1, 28.2), far in excess of the corresponding values for an 11-fault-tolerant 125-node 5-ary K-cubes or 15625-node 5-ary C-cubes. For these reasons we pursue holistic solutions to 4.1.1.

Radius and diameter are related, but somewhat different measures of latency.[42] Nodes with minimum eccentricity (equal to the radius) are *central*. Nodes with maximum eccentricity (equal to the diameter) are *peripheral*. A tendency among contemporary point-to-point bus standards (such as Firewire[37, 52]) is to prescribe connectivity using the fewest number of edges; *i.e.*, to force a quorum that comprises a *tree*.[50] Where a central master marshals bus traffic (as is the case with Firewire), it makes sense to assign mastery to the *root* of the tree, and (in the interest of minimizing latency) to make this root a central node of the quorum. Unfortunately, such predilection for configuring a tree is detrimental, for at least two reasons:

i) Trees *overconstrain* what we need to build (*cf.* paragraph preceding requirement 3.7, p. 12)

ii) Insisting on configuration of a tree discards edges that could be used to carry signals. This a) artificially limits throughput, and b) has the practical effect of re-initializing the system whenever a fault is detected.

Since a path is a special case of a tree, remark (i) pertains in particular to the $P_{n\text{-}i}$'s configured from the chordal graphs of [Hayes 1976]. In addition, the tree architectures considered by [Hayes 1976] are tolerant to at most one fault, and trees configured from these are balanced. By comparison to the problem we wish to consider, this is overconstrained (our trees need not be balanced, we wish to tolerate more than one fault). There are as well differences with a number of other works. [Kwan and Toida 1981] consider tolerance to one and two faults for balanced trees, and whose every level represents a potentially different type of processor. [Dutt and Hayes 1990] use vertex covering to design balanced j-ary trees that are optimal when $f < j$.[53] In terms of hardware, algorithms, and human effort, these examples illustrate how trees overconstrain what we are to build.

Though not explicated in 4.1.1, *throughput* tends to improve as latency improves.[54] For example, the worst-case source-destination throughput for a minimum size f-tolerant graph architecture containing $i \le f$ faults equals $f + 1 - i$ times the edge capacity. As illustrated in Figure 20, STAArchitecture uses max-flow algorithms to calculate throughput of a faulted instance. To maximize *average* throughput (single source and destination, or aggregate overall [Sampels 1997]), we want to retain as many edges as possible. Insisting that our quorum be a tree (iia) defeats this goal; it also imposes unnecessary complexity (iib) on *algorithms* for diagnosis and configuration. This latter point is underscored by Figures 22 through 24.

50. A graph T of order n is a *tree* if and only if T is connected and cycle-free; equivalently, T is connected and has minimum size n-1 ([Chartrand and Lesniak 1986], Chapter 3). T is said to *span* H if T and H have the same vertices and every edge of T is an edge of H. Equivalent conditions for H to be connected: (*cf.* page 23): i) every pair of vertices is connected by at least one path; ii) H is spanned by a tree.
51. Here f is assumed to be odd. The quorum radii may in fact be equal, but only in a finite number of instances. In one dimension, K-cube-connected cycles are known as *secant* graphs ([LaForge 1999 JPL D-16485], Sec. 3.6).
52. *Cf.* Firewire COTS intellectual property as a ticking box, Sec. 3.13, p. 15.
53. Still other works treat configuration of balanced trees in either a probabilistic context, or with respect to VLSI layout area and maximum wirelength (e.g., [Chen and Upadhyaya 1993]).
54. This is analogous to a programming rule of thumb whose origins are rooted in the theory of computation ([Hopcroft and Ullman 1979] Chap. 12): the space complexity of a program is bounded by its time complexity.

Distributed Diagnosis and Configuration Algorithm A_{tree}   % Configure a tree from a cycle

```
1)  Enable all ports                                                   % N.B. runs on $u_i$; initial port
2)      except those between $(u_{n-1}, u_0)$, $(u_{\lfloor n/2 \rfloor}, u_{[\lfloor n/2 \rfloor + 1] \bmod n})$   % disable values in ROM
3)  Initial bus reset                                                  % Bus reset (not command reset)
4)  For each of $u_i$'s enabled ports in $P_{0, \lfloor n/2 \rfloor}$              % Have at most one fault;
5)      If    test $(u_i, u_{[i-1] \bmod n})$ fails                         %  hence at most 2 buses are formed
6)      then  $u_i$ marks $u_{[i-1] \bmod n}$, disables its port to $u_{[i-1] \bmod n}$   % Record results of failed test
7)            $u_i$ issues a bus reset                                  %  and disable immediately
8)      If    test $(u_i, u_{[i+1] \bmod n})$ fails                         % Performing a bus reset
9)      then  $u_i$ marks $u_{[i+1] \bmod n}$, disables its port to $u_{[i+1] \bmod n}$   %  guarantees two leaves
10)           $u_i$ issues a bus reset
11) Propagate the marked status of each node throughout bus           % Get info to least one of $u_0$, $u_{\lfloor n/2 \rfloor}$
12) $u_0$ disables its port to $u_1$; $u_1$ disables its port to $u_0$              %  Switch to complementary bus
13) $u_{\lfloor n/2 \rfloor}$ disables its port to $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$; $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$ disables its port to $u_{\lfloor n/2 \rfloor}$
14) $u_0$ enables its port to $u_{n-1}$; $u_{n-1}$ enables its port to $u_0$
15) $u_{\lfloor n/2 \rfloor}$ enables its port to $u_{[\lfloor n/2 \rfloor + 1] \bmod n}$; $u_{[\lfloor n/2 \rfloor + 1] \bmod n}$ enables its port to $u_{\lfloor n/2 \rfloor}$
16) $u_0$ and $u_{\lfloor n/2 \rfloor}$ issue bus reset                              % Node insertion/ deletion
17) For each of $u_i$'s enabled ports in $\overline{P}_{\lfloor n/2 \rfloor, 0}$              % Have at most one fault;
18)     If    test $(u_i, u_{[i-1] \bmod n})$ fails                         %  hence at most 2 buses are formed
19)     then  $u_i$ marks $u_{[i-1] \bmod n}$, disables its port to $u_{[i-1] \bmod n}$   % Record results of failed test
20)           $u_i$ issues a bus reset                                  %  and disable immediately
21)     If    test $(u_i, u_{[i+1] \bmod n})$ fails                         % Performing a bus reset
22)     then  $u_i$ marks $u_{[i+1] \bmod n}$, disables its port to $u_{[i+1] \bmod n}$   %  guarantees two leaves
23)           $u_i$ issues a bus reset
24) Propagate the marked status of each node throughout bus           % Get info to least one of $u_0$, $u_{\lfloor n/2 \rfloor}$
25) If    $u_1$ is not marked by $u_0$
26) then $u_0$ enables its port to $u_1$
27) If    $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$ is not marked by $u_{\lfloor n/2 \rfloor}$               % Have at most one fault;
28)         and some other node is marked                              %  hence $u_0$, $u_{\lfloor n/2 \rfloor}$, if not faulty,
29) then $u_{\lfloor n/2 \rfloor}$ enables its port to $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$               %  has status of marked nodes
30) If    $u_0$ is not marked by $u_1$
31) then $u_1$ enables its port to $u_0$
32) If    $u_{\lfloor n/2 \rfloor}$ is not marked by $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$
33)         and some other node is marked
34) then $u_{[\lfloor n/2 \rfloor - 1] \bmod n}$ enables its port to $u_{\lfloor n/2 \rfloor}$
35) $u_0$ and $u_{\lfloor n/2 \rfloor}$ issue bus reset                              % Final configuration
```

Distributed Diagnosis and Configuration Algorithm $A_{\text{connected}}$ % Configure a quorum from a cycle

```
                                                                       %  N.B. runs on $u_i$.
1)      If    test $(u_i, u_{[i+1] \bmod n})$ fails                         %  Ports initially enabled.
2)      then  $u_i$ disables its port to $u_{[i+1] \bmod n}$                    %  Test and configure
3)      If    test $(u_i, u_{[i-1] \bmod n})$ fails                         %    clockwise
4)      then  $u_i$ disables its port to $u_{[i-1] \bmod n}$                    %    then counterclockwise
```

Figure 22: As the code for $A_{\text{connected}}$ indicates, permitting cycles simplifies diagnosis and configuration. Example is for configuration from C_n, the minimum size 1-fault-tolerant graph architecture on n nodes.

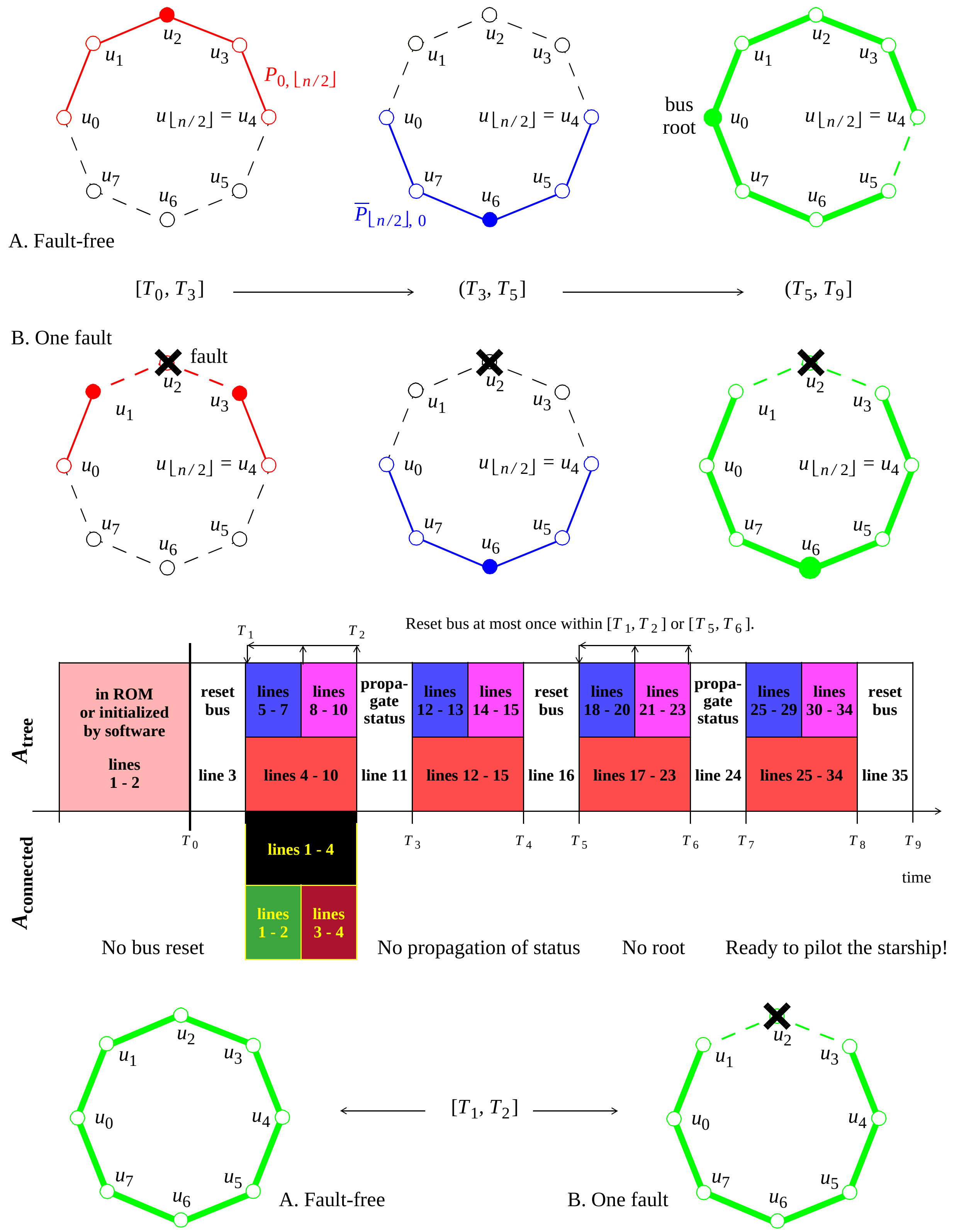


Figure 23: *Dynamic* reasons to permit cycles in the course of diagnosis and configuration. Doing so reduces both *synchronization interfaces* and *running time*: $T_1 \le A_{\text{connected}} \le T_2$, while $T_0 \le A_{\text{tree}} \le T_9$. Consistent with Figure 22, the parallel-series event timeline illustrates configuration from C_n; $n = 8$.

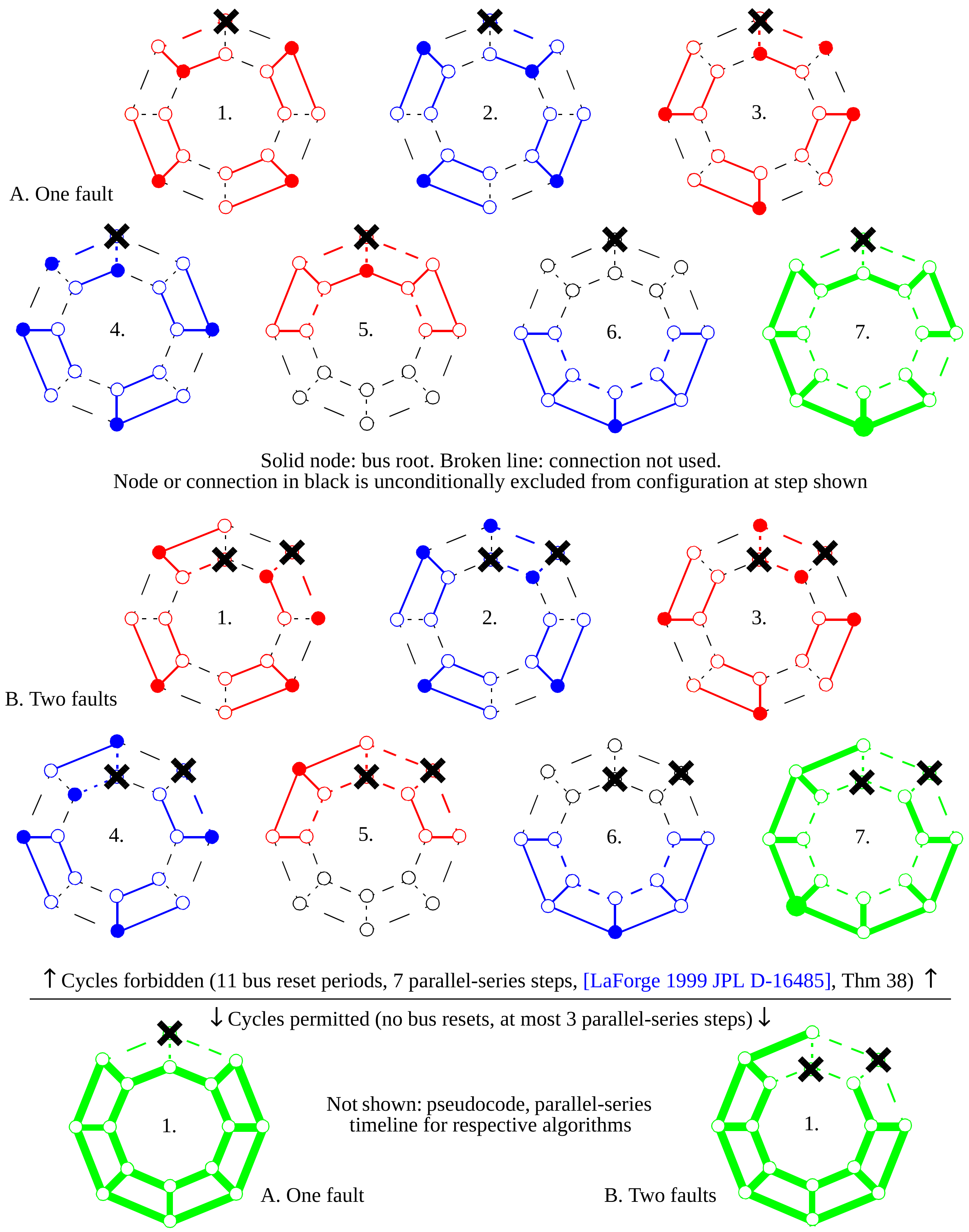

A. One fault
1.
2.
3.
4.
5.
6.
7.
Solid node: bus root. Broken line: connection not used.
Node or connection in black is unconditionally excluded from configuration at step shown
B. Two faults
1.
2.
3.
4.
5.
6.
7.
↑ Cycles forbidden (11 bus reset periods, 7 parallel-series steps, [LaForge 1999 JPL D-16485], Thm 38) ↑
↓ Cycles permitted (no bus resets, at most 3 parallel-series steps) ↓
1.
Not shown: pseudocode, parallel-series timeline for respective algorithms
A. One fault
B. Two faults
1.

Figure 24: Another demonstration: although the structure of a tree is less complicated than the connected component it spans, extracting such a tree can complicate matters. As a function of fault tolerance, the complexity of diagnosis and configuration increases if (sub-)quorum cycles are forbidden. Compared here: configuration from a 16-node 2-fault-tolerant binary K-cube-connected cycle (*cf.* Figure 19).

Continuing from page 27, our pronounced elaboration of points (i) and (ii) is in part directed at stemming the cultural tide of trees as *desiderata* for quorums.[55] While it is true that trees foster simplification of routing algorithms,[56] unbridled application of this fact undercuts performability. In summary:

4.2.1 The target quorum for self-healing architectures should not be limited to a tree that spans healthy nodes, but rather should include *all* edges between healthy nodes.

Contrasting trees with connected components, we are now in a position to distinguish radius and diameter. Denote by H an arbitrary quorum induced by deleting i nodes of an n-node $(f+1)$-connected graph G. If $0 \le i \le f$ then this quorum is guaranteed to be connected, and therefore possesses one or more spanning trees.[50] Any tree has at most two central vertices, and they (it) always lie(s) at the intersection of maximum length path(s). An immediate corollary is that the diameter of a tree is either twice its radius, or twice its radius minus one ([LaForge 1999 JPL D-16485] Thm 1, Cor. 1.1). The power of this observation is bolstered by a theorem of [Ore 1962]: for every node u of a connected graph H, there exists a spanning tree of H that is distance-preserving from u. Moreover, we can compute, on a Turing machine equivalent and in time $O(n(n+e))$, a spanning tree having minimum radius ([LaForge 1999 JPL D-16485] Thm 36).[48] Taken together, these results imply that a there is a spanning tree of H whose radius is identical to that of H … the best we could hope for. On the other hand, the diameter of this tree is equal to, or one less than, twice the radius … *i.e.*, the worst-case latency is about as bad as it gets.[42] By contrast, and as depicted in Figure 21, the quorum itself may have a diameter much smaller than twice the radius, even approaching the value of the radius itself. The highly performable structures synthesized by STAArchitecture exhibit this feature. For these reasons our analysis emphasizes radius.

Through the end of Section 4.5 we assume: i) faults have been correctly diagnosed; ii) the outcome of this diagnosis is passed to a configuration *algorithm* (which may be distributed, *cf.* Figures 22 through 24); iii) nodes (but not edges) may be faulty (*i.e.*, deleted). Since edge connectivity is no less than vertex connectivity, item (iii) does not materially affect our analysis; however, allowing the deletion of edges can change the sharpness of our results for radius and diameter.[44] Refer Table 7. STAArchitecture's candidates for configuration architectures are members of the set $\mathcal{G}^{+}_{n,f,k}$ of minimum size $(f+1)$-connected graphs of order n whose quorums, induced by deletion of up to f vertices, have radii at most k. For given n and f, we naturally wish to assure that k is the exact minimum, in which case we write $\mathcal{G}_{n,f}$, perhaps with an extra subscript k. We denote the corresponding radius by $\rho(n, f)$. Although the general solution to this problem appears to be unknown,[57] we can enumerate $\mathcal{G}_{n,0,k=2}$, $\mathcal{G}_{n,1,k=\lfloor n/2 \rfloor}$, and $\mathcal{G}_{n,n\text{-}2,k=2}$; that is, $\rho(n, 0) = 2$, $\rho(n, 1) = \lfloor n/2 \rfloor$, and $\rho(n, n\text{-}2) = 1$. For other values of f, we provide upper and lower bounds on $\rho(n, f)$, and give sets $\mathcal{G}^{+}_{n,f,k}$ whose induced quorums have $O(\log n)$ radii.

55. The complexity illustrated by Figures 22 through 24 goes deeper than first appears. If one insists that each subquorum used to form a tree be cycle-free, then to cover all edges of the embedding architecture (*i.e.*, test every possible pairwise connection), we must *factor* the corresponding graph into *forests* of trees. The union of these forests comprises the embedding architecture, and pairs of forests are edgewise disjoint. Under the constraint of cycle-free configuration, the extent to which diagnosis and configuration can be parallelized is *at least* the minimum number of forests in any such factorization. This number is known as the *aboricity* of the graph, and can be calculated (albeit somewhat awkwardly) by applying a result of Nash-Williams: Let e_m be the maximum size of any subgraph of order m; the arboricity is the largest value of $\max_{1 \le m \le n} \left\lceil \frac{e_m}{m-1} \right\rceil$ ([Bollabás, 1978], Thm 5.8). By contrast, a generalization of our connected component algorithm (Figure 22) executes in a number of serial steps that is at most the maximum degree of a node in the graph architecture. [LaForge 1994] illustrates application of matching and Hamiltonian cycles to parallel scheduling of tests among nodes in locally spared arrays.

56. As long as everything works properly, the simplification exploits the following property (which is, in fact, an equivalent definition of a tree): between any two nodes there is a unique path (*cf.* [Anderson 1998] Chap 13-17).

Symbol	Significance
$\lceil x \rceil$; $\lfloor x \rfloor$	Ceiling (least integer no less than x); floor (greatest integer no greater than x)
$<u,v>$; $<P>$	Graph distance between vertices u and v; length of path P
$O(g(n))$; $\Omega(g(n))$	Set of functions no greater *resp.* no less than $c \cdot g(n)$, for $n > k$, constants c, k
$o(g(n))$; $\omega(g(n))$	Set of functions $h(n)$ such that $\lim_{n \to \infty} h/g = 0$ *resp.* $\lim_{n \to \infty} g/h = 0$
$\Theta(g(n))$	Intersection of $O(g(n))$ and $\Omega(g(n))$
$B_j^C(d,i)$	Number of vertices at graph distance i from any vertex in C_j^d
C_n; C_j^d	n-vertex cycle; d-dimensional j-ary C-cube
e, $e_K(d,j)$; $e_K(d,j,n)$; $e_C(d,j)$	Size (number of edges) of a graph; of a K_j^d; of a $K_j^d(n)$; of a C_j^d
f, f_{frac}	Number, fraction f/n of faulty elements (deleted vertices) that can be tolerated
G	Graph, often one that represents the configuration architecture
$\mathcal{G}^+_{n,f,k}$	Set of minimum size $(f+1)$-connected graphs of order n whose quorums, induced by deletion of up to f vertices, have radii at most k
$\mathcal{G}_{n,f}$, $\mathcal{G}_{n,f,k}$	Set $\mathcal{G}^+_{n,f,k}$ that minimizes the maximum radius k
H ; T	Quorum induced by deleting vertices from G; tree, often one that spans H
$K_n = K_n^1$; K_j^d	n-vertex clique; d-dimensional j-ary K-cube
$K_j^d(n)$; $K_{m\text{Þ}j}^d$	d-dimensional j-ary K-cube-connected cycle on n *resp.* $m \cdot j^d$ vertices
n; $n_K(d,j)$; $n_C(d,j)$	Order (number of vertices) of a graph; of a K_j^d; of a C_j^d
$\rho(n, f)$; $\rho^{-}_{\text{Thm 6}}$	Maximum radius among quorums induced by $\le f$ faults; Moore bound of ineq. (1)
P_n; S_n	n-vertex path; n-vertex star
$V_j^C(d,i)$	Number of vertices graph distance at most i from any vertex in C_j^d

Table 6: Notation.

We can break our analysis into four stages: i) bound the maximum radius $\rho(n, i)$ of any quorum, as a function of the number i of vertices deleted; ii) find the maximum among these maxima $\rho(n, i)$, for $0 \le i \le f$; iii) convert to bounds on the diameter; iv) compare the corresponding results for different structures to each other, as well as to a general lower bound $\rho^{-}_{\text{Thm 6}}$ on the radius. The latter

$$\rho(n, f) \ge \left\lceil \log_f \left\lceil \frac{n(f-1)+3}{f+2} \right\rceil \right\rceil, \; 1 < f < n\text{-}2 \qquad (1)$$

is obtained by maximizing an inequality that takes into account i faults. As derived for Theorem 6 of [LaForge 1999 JPL D-16485], that is, $\rho(n, i)$ is at least

57. The closest body of work seems to be related to the function $\varphi(n,d_0,d,f)$, introduced by [Murty and Vijayan 1964]. Here j counts the minimum number of edges in an n-node graph with diameter at most d_0, such that deletion of any f of the vertices induces a graph of diameter at most d. Even for this relatively well-studied problem, results are confined primarily to the cases $d \le 4$, $f = 1$ or $d_0 = 2$ ([Bollabás, 1978], Chapter IV, Sections 2 and 3). Moreover, our formulation differs in that we *fix* the number of edges at $\lceil (f+1)n/2 \rceil$, and then ask for the minimum diameter or radius achievable in the induced quorum.

$$\left\lceil \log_f \left[\frac{(n-i-1)(f-1)+f+1+[n(f+1) \bmod 2]}{f+1+[n(f+1) \bmod 2]} \right] \right\rceil \qquad (2)$$

For graphs with maximum *degree* (as opposed to *connectivity*) $f + 1$, the independently obtained (1) is equivalent to the bound attributed to Moore [Sampels 1997], and we will continue this custom.[58] In particular, any minimum size (f+1)-tolerant graph that achieves equality in (1) is optimal. Refer to the last row of Table 7. An n-node clique K_n (that is, a graph of order n and maximal size ½·n(n-1)) is tolerant to

Fault tolerance f	Graph architectures	Maximum of quorum radii $\rho(n, i)$, $0 \le i \le f$		Maximum radius of quorum divided by lower bound $\rho^-_{\text{Thm 6}}$	References
		At least	At most		
0	$\mathcal{G}_{n,0}$ uniquely the set of n-vertex stars S_n	1		Exactly best possible	Table 15
1	$\mathcal{G}_{n,1}$ uniquely the set of n-vertex cycles C_n	$\lfloor n/2 \rfloor$		Exactly best possible	Table 15
2	$\mathcal{G}^+_{n,2}$ includes 1-dimensional binary K-cube-connected cycles $K_2^{\,1}$(n = 2m+1), $m \ge 2$	1 if n = 5 else $1 + \lfloor \lfloor n/2 \rfloor /2 \rfloor$	$1 + \lfloor \lceil n/2 \rceil /2 \rfloor$	Don't know	Table 13 Thm 6, discussion on p. 40
2·[log$_j$ n] -1 = 2d - 1,	$\mathcal{G}^+_{n,2d-1}$ includes d-dimensional j-ary C-cubes $C_j^{\,d}$; $j \ge 5$	$\lfloor j/2 \rfloor \cdot \log_j n$	$\lfloor j/2 \rfloor \cdot (\log_j n) + \lceil j/2 \rceil - 1$	Definitely *not* best possible: ratio diverges to ∞ as $n \to \infty$	Table 10 Thms 6 Thms 6, 7 Cor 7.1
[(j-1)·log$_j$ n] -1 = (j-1)·d - 1	$\mathcal{G}^+_{n,(j-1)d-1}$ includes d-dimensional j-ary K-cubes $K_j^{\,d}$	$\log_j n$	$1 + \log_j n$	As $n \to \infty$: approaches best possible whenever $d \in o(j)$ and $m \in o(d)$ or d and m bounded. Within 1+q+qr+r of best possible whenever $\left\lceil \frac{m}{2} \right\rceil + 1 \le qd$ and $\ln d \le r \ln j$, for least upper bounds q, r.	Tables 11, 13, and 14 Cor 5.1 Cor 5.2 Thm 6
(j-1)·log$_j$ (n/2) = (j-1)·d	$\mathcal{G}^+_{n,(j-1)d}$ includes d-dimensional j-ary K-cube-connected edges $K_{2\cdot j}^{\,d}$, $j \ge 3$	2 if d = 1			
		$1 + \log_j (n/2)$	$2 + \log_j (n/2)$		
1 + (j-1)·log$_j$ (n/m) = (j-1)·d + 1	$\mathcal{G}^+_{n,(j-1)d+1}$ includes d-dimensional j-ary K-cube-connected cycles $K_{m\cdot j}^{\,d}$, $m \ge 3$	$1 + \lfloor m/2 \rfloor$ if d = 1			
		$\lfloor m/2 \rfloor + \log_j (n/m)$	$1 + \lfloor m/2 \rfloor + \log_j (n/m)$		
n-2, n-1	$\mathcal{G}_{n,n-2,1}$, $\mathcal{G}_{n,n-1,1}$ uniquely the set of n-vertex cliques K_n	1		Exactly best possible	Table 15 Thm 6

Table 7: STAArchitecture applies knowledge about the maximum quorum radius. Entries in red indicate deliverables derived for this Phase I effort. Citations in blue refer to [LaForge 1999 JPL D-16485].

58. In contrast to inequality (1), Moore's bound is concerned with the maximum order of a graph with bounded diameter and degree, in the absence of faults. Both results make use of arguments that minimize the height of a spanning tree, with application of the formula for summing of a geometric series.

$f = n$-2 or $f = n$-1 faults, has minimum size, and delivers a quorum radius or diameter that is at most one. Substituting the latter gives equality in (1), hence K_n matches the Moore bound and is optimal. Furthermore, with respect to our cost criteria, K_n is the *unique* optimum graph that is tolerant to $f = n$-2 or $f = n$-1 faults (Table 15).

A weaker but still strong criterion asserts the optimality of a family of graphs if, as n approaches infinity, the maximum quorum radius is within a constant factor of the Moore bound. Referring to the next-to-last three rows of Table 7, we see that such *ratioed asymptotic* optimality is indeed achieved by members of the K-cube family. The mainstay of this family, a *d-dimensional Gray-coded j-ary K-cube* K_j^d is recursively constructed as follows. K_j^0 is a lone node labeled with the null string. For K_j^d we i) make j copies of $K_j^{d\text{-}1}$; ii) join with an edge nodes u and v (from different copies of $K_j^{d\text{-}1}$) if and only if u and v have with identical labels; iii) prepend i to the label of each node of the ith copy of $K_j^{d\text{-}1}$. Note that K_j^1 is just the clique K_j whose nodes have been labeled from 0 to j-1. Figure 25 illustrates binary and ternary K-cubes in 3 *resp*. 2 dimensions. Very few graphs are known to match the Moore bound [Sampels 1997], and our results establishing this for the K-cube family appear to be new. From a practical standpoint, the K-cube family delivers asymptotically minimum quorum radii whenever fault tolerance is on the order of $n^{1/d} \log n$.

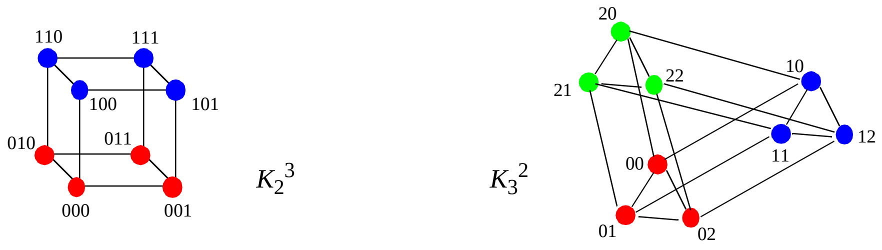


Figure 25: Gray-code labeling of a three-dimensional K$_2$-cube and a two-dimensional K$_3$-cube.

Somewhat surprisingly, there are well-studied (f+1)-tolerant graph architectures which are mistakenly believed to deliver optimal, or near optimal, quorum radii. As indicated by Table 7, this is indeed the case with C-cubes. Often referred to in the literature as a "hypercube" or simply a "cube", a *labeled d-dimensional j-ary C-cube* C_j^d is constructed as follows. For $j = 2$: C_2^d is a d-dimensional binary K-cube K_2^d (equivalently, a (d-1)-dimensional binary K-cube-connected edge $K_{2\cdot 2}^{d\text{-}1}$); for $j = 4$: C_4^d is a K_2^{2d} (proof by induction); binary cubes are characterized Section 3.3 of [LaForge 1999 JPL D-16485]. For $j > 2$: C_j^0 is a single unlabeled node. 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. Note that a <u>one</u>-dimensional j-ary C-cube C_j^1 is the same as a j-node <u>zero</u>-dimensional j-ary K-cube-connected cycle $K_{j\cdot j}^0$. In general, to construct C_j^d we i) make j copies of $C_j^{d\text{-}1}$; ii) prepend i to the label of each node of the ith copy of $C_j^{d\text{-}1}$; iii) connect with an edge vertices u and v (from different copies of $C_j^{d\text{-}1}$) if and only if the modulo j difference in the high order digits of the labels on u and v equals ±1, <u>and</u> the low order d-1 digits are identical. Alternatively, we can reserve d digits for the label on each node, thus giving to rise a construction that is independent of the order in which dimensions are populated. Figure 25 illustrates 4-ary and ternary C-cubes in 2 *resp*. 3 dimensions. Note that, since a cycle on three nodes is also a three-node clique, $C_3^d = K_3^d$ (equivalently, a (d-1)-dimensional ternary K-cube-connected cycle $K_{3\cdot 3}^{d\text{-}1}$); K_3^d's are characterized by Section 3.3 of [LaForge 1999 JPL D-16485]. It suffices therefore to consider dimensions $d \geq 2$ and radices $j \geq 5$, and such is the focus of our comparison.

The volume of literature concerning C-cubes exceeds perhaps that of any other structure studied in fault tolerance or networks (to scratch the tip of the iceberg: [Agarwal 1991], [Armstrong and Gray 1981], [LaForge 1994]). For this reason, it is especially surprising that K-cubes and their relatives are preferred to C-cubes. Quantitatively, this is due to: 1) the radius of a C-cube quorum exceeding the diameter of the comparable K-cube having identical fault tolerance (Thm 6); 2) there being *no* relation such that, as

$n_C = j^d \rightarrow \infty$, the ratio of the C-cube quorum radius to the Moore bound does *not* diverge; *i.e.*, this ratio must approach infinity. With respect to both criteria, that is, C-cubes are sub-optimal. Moreover, when scaling is such that K-cubes match the Moore bound, C-cubes diverge from the optimal quorum radius (Cor 7.1). In this ratioed asymptotic sense, K-cubes are optimal, whereas C-cubes are sub-optimal. Figure 21 illustrates these observations for the lowest radix (5) where C-cubes and K-cubes differ. For $j > 4$ it is impossible to find a C-cube whose fault tolerance and number of nodes equals that of a K-cube, and so we have compared the respective structures having identical fault tolerance. The fractional fault tolerance of C-cubes is less than that of K-cubes, and so the performability comparison is conservative. Sections 4.3 and 4.5 establish these results in detail.

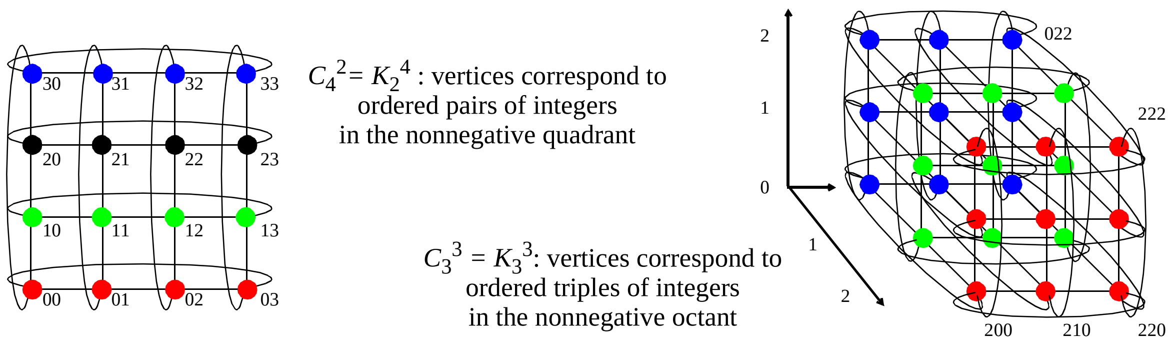

30
31
32
33
20
21
22
23
10
11
12
13
00
01
02
03
$C_4{}^2 = K_2{}^4$: vertices correspond to ordered pairs of integers in the nonnegative quadrant
$C_3{}^3 = K_3{}^3$: vertices correspond to ordered triples of integers in the nonnegative octant
2
1
0
1
2
022
222
200
210
220

Figure 26: Labeling and connectivity for a C_4-cube and C_3-cube = $K_3{}^3$ in two *resp.* three dimensions.

4.3 STAArchitecture Reflects New Results About Quorums from C-Cubes [59]

As with K-cubes, it is useful to know salient properties of C-cubes. Some (but not all) of these properties are listed in [Zargham 1996] (p. 204). Recalling that the radix j is greater than four, let us establish results pertaining to these properties. By step (i) on the preceding page, $C_j{}^d$ contains j copies of $C_j{}^{d\text{-}1}$; therefore the order $n_C(d, j)$ of $C_j{}^d$ equals $j \cdot n_C(d\text{-}1, j)$. Subject to the initial condition $n_C(0, j) = 1$, verify that the unique solution of this recurrence relation is the same as that for the number of vertices in a j-ary K-cube:

$$n_C(d, j) = j^{\,d} \tag{3}$$

By step (iii) on the preceding page, the degree of a vertex[60] in $C_j{}^d$ equals its degree in $C_j{}^{d\text{-}1}$ plus 2, the number of edges that connect it to vertices with the same labels in neighboring copies of $C_j{}^{d\text{-}1}$. Subject to the initial condition of zero edges in $C_j{}^0$, the degree of each vertex in $C_j{}^d$ is therefore $\quad 2d \quad$ (4)

Summing (4) over all $j^{\,d}$ vertices counts every edge twice. Hence the number $e_C(d, j)$ of edges in $C_j{}^d$ is

$$e_C(d, j) = d \cdot j^{\,d} \tag{5}$$

As is the case with K-cubes (as well as edges and cycles of K-cubes), C-cubes are *vertex symmetric*.[61] Moreover, and as illustrated by Figure 26, the vertices of $C_j{}^d$ are in one-to-one correspondence with

59. We use a "C" to preface the term for a cube $C_j{}^d$ that is based on cycles, as opposed to a clique-based (K-)cube; with respect to the latter, the K derives ([LaForge 1999 JPL D-16485] Sec 3.7) from notation for a j-vertex clique K_j.
60. Reflecting prevalent terminology in extremal graph theory, in this section we prefer "vertex" over "node".[43]
61. Loosely speaking, a graph is *vertex symmetric* if the perspective is the same from every vertex. More precisely, a graph G is *vertex-symmetric* if the group $A(G)$ of graph automorphisms of G acts transitively on V; *i.e.*, for any $v, w \in V$, there is a graph automorphism $\alpha \in A(G)$ such that $\alpha(v) = w$ ([Biggs 1993] p. 115).

ordered d-tuples, each of whose coordinates is a nonnegative integer. This suggests that, if two vertices $u = (u_{d-1}, \ldots, u_0)$ and $v = (v_{d-1}, \ldots, v_0)$ are sufficiently close, their distance should be given by the L_1 metric (also known as the *city block*, or *Manhattan* metric):

$$\langle u, v\rangle_1 = \sum_{k=0}^{d-1} |u_k - v_k| \tag{6}$$

This tendency is born out by the L_1 "modulo j" metric of (7). If u and v are vertices of K_j^d, Gray-code labeled according to steps (i) – (iii) on page 34, then $\langle u, v\rangle$ equals the number of digits where the respective labels for u and v are different ([LaForge 1999 JPL D-16485] Thm 7). The analog for C-cubes:

Theorem 1. If u and v are vertices of C_j^d, labeled according to steps (i) – (iii) on page 34, then

$$\langle u, v\rangle_{\text{mod } j} = \sum_{k=0}^{d-1} \min(|u_k - v_k|, j - |u_k - v_k|) \tag{7}$$

Proof. Regard arbitrary vertices u and v in C_j^d. Since C_j^d is vertex symmetric, we can assume without loss of generality that $u = (0, \ldots, 0) = \mathbf{0}$. By step (iii) on page 34, we must traverse at least $\min(v_k, j-v_k)$ edges along the ith axis. Thus the distance from $\mathbf{0}$ to v is at least (7). Further, and again by the construction on page 34, this bound is achieved by traversing v_k edges in the positive direction of the ith axis (if $v_k \le j-v_k$) or (if $v_k > j-v_k$) by traversing $j-v_k$ edges in the negative direction of the ith axis. ❐

Equation (7) is maximized when the respective terms in the summation are maximized. That is, when $v_k = \lfloor j/2 \rfloor$, for all k ranging between 0 and d - 1. It immediately follows:

Corollary 1.1. The radius and diameter of C_j^d are identically $d \cdot \lfloor j/2 \rfloor$.

Corollary 1.1 addresses the case of a C_j^d without faults. To derive a lower bound on radius, consider the number $B_j^C(d, i)$ of integer lattice points <u>on</u> the surface of, as well as the total number $V_j^C(d, i)$ <u>in</u>, a closed ball of L_1 modulo j radius i. By Corollary 1.1 and equation (7), we know that $V_j^C(d, d\cdot\lfloor j/2 \rfloor) = j^d$ (8)

For the sake of visualization assume that j is odd; translate the labels of C_j^d so that the point $((j-1)/2, \ldots, (j-1)/2)$ becomes the origin. By (7), any point v in the ball of interest belongs to an L_1 ball centered at the new origin, as long as all of the (translated) coordinates of v satisfy $v_k \le (j-1)/2$. Let us establish the volume and surface area of such a ball. If the radius i equals 0 then the ball contains just the origin, which is also on the surface in the sense that it is the number of points exact distance 0 from the center. Adopting the latter definition:

$$B_j^C(0, 0) = V_j^C(d, 0) = 1 \tag{9}$$

At the outset it is not clear what meaning we should accord the surface area of zero-dimensional ball with positive radius. However, if we hold strictly to the definition used for (9) then the surface area of a zero-dimensonal ball equals zero whenever $i > 0$:

$$B_j^C(0, i > 0) = 0 \tag{10}$$

whence $$V_j^C(0, i) = 1 \tag{11}$$

Refer to Figure 27. Equations (9), (10), and (11) are consistent with the one-dimensional case $B_j^C(0, i) = 2$ and $V_j^C(0, i) = 2i+1$ (which could have served as boundary conditions) as well as with the respective recurrences:

$$B_j^C(d,i) = B_j^C(d-1,i) + 2 \sum\nolimits_{k \le 0 \le i-1} B_j^C(d-1,k) = B_j^C(d-1,i) + B_j^C(d-1,i-1) + B_j^C(d,i-1) \tag{12}$$

$$V_j^C(d,i) = V_j^C(d\text{-}1,i) + 2 \sum_{k \le 0 \le i\text{-}1} V_j^C(d\text{-}1,k) = V_j^C(d\text{-}1,i) + V_j^C(d\text{-}1,i\text{-}1) + V_j^C(d,i\text{-}1) \qquad (13)$$

To obtain the righthand relation we have recursively applied the lefthand side to a split sum. Table 8 illustrates computation of B_j^C and V_j^C , and serves as the analog to tabulations of the surface area of balls in K-cubes, K-cube-connected cycles, and K-cube-connected edges (*cf.* [LaForge 1999 JPL D-16485] Tables 8, 10, and 13) .

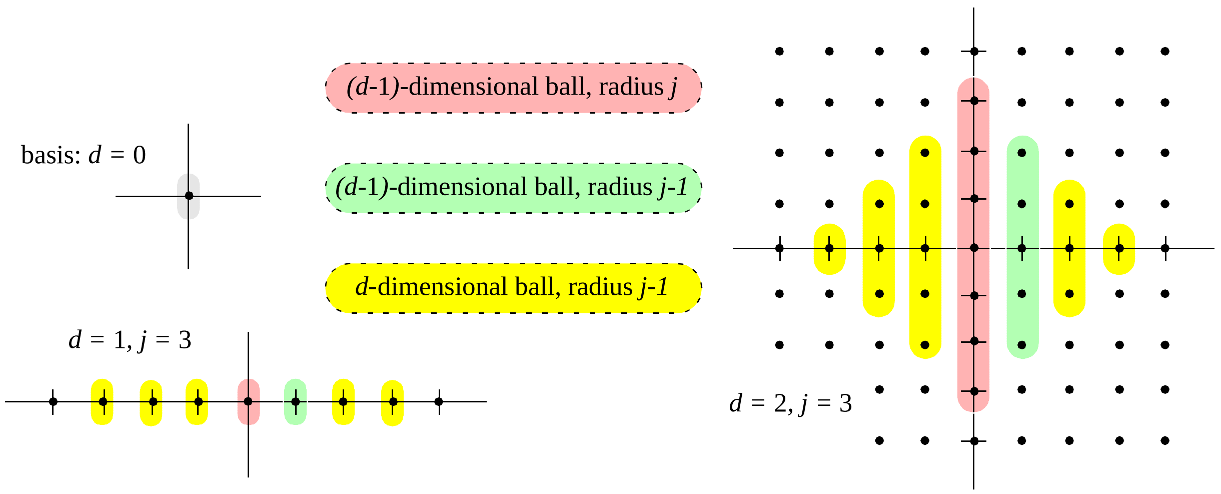

basis: d = 0
(d-1)-dimensional ball, radius j
(d-1)-dimensional ball, radius j-1
d-dimensional ball, radius j-1
d = 1, j = 3
d = 2, j = 3

Figure 27: Balls in the L_1 metric: recursive composition and enumeration of volume and surface area.

Notice that the recurrence (12) for B_j^C is the same as that (13) for V_j^C, but boundary condition (10) for B_j^Cdiffers from that (11) for V_j^C. As a result, and as illustrated in Table 8, B_j^C is asymmetric, while V_j^C is a symmetric function of d and j. Let us use combinatorial means to solve for V_j^C. Again we focus on balls centered at u = **0** in the translated coordinate system, and restrict the absolute value of each coordinate of v to a value no greater than (j-1)/2 .

↓ d	$B_j^C(d,i)$								$V_j^C(d,i)$							
→ i	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	1	2	2	2	2	2	2	2	1	3	5	7	9	11	13	15
2	1	4	8	6	12	16	20	24	1	5	13	25	41	61	85	113
3	1	6	18	38	66	102	146	198	1	8	25	63	129	231	377	575
4	1	8	32	88	192	360	608	952	1	10	41	129	321	681	1289	2241
5	1	10	50	170	450	1002	1970	3530	1	12	61	231	681	1683	3653	7183
6	1	12	72	292	912	2364	5336	10836	1	12	85	377	1289	3653	8989	19825
7	1	14	98	462	1666	4942	12642	28814	1	12	113	575	2241	7183	19825	48639

Table 8: B_j^C and V_j^C count the number of vertices on the surface of, *resp.* included in, a closed ball encompassing integer lattice points, each of whose distance from the center is no greater than $\lfloor j/2 \rfloor$ = 7. The ball has integer L_1 radius i, and is centered at a point whose coordinates correspond to a label in C_j^d.

Consider the 2^d-tant comprising all strictly positive coordinates included in a ball of L_1 radius i. The number B_j^{C+} of positive integer lattice points on the surface of this ball equals the number of solutions to

$$i = \sum_{k=0}^{d-1} v_k \tag{14}$$

Equation (14) has ordinary generating function: $$\left[\frac{x}{1-x}\right]^d = \left[x\sum_{k=0}^{\infty} x^k\right]^d = x^d \sum_{k=1}^{\infty} \binom{d+k-1}{k} x^k \tag{15}$$

By Chapter 6 of [Tucker 1984], $B_j^{C+}(d,i)$ is the coefficient of x^i in (15): $$B_j^{C+}(d,i) = \binom{i-1}{i-d} = \binom{i-1}{d-1} \tag{16}$$
where the righthand side makes use of the symmetry of binomial coefficients. Summing over all i yields the volume of intersection of the ball with the strictly positive 2^d-tant:

$$\begin{aligned} V_j^{C+}(d,i) &= \sum_{k=d}^{i}\binom{k-1}{d-1} = 0 + \binom{d-1}{d-1} + \sum_{k=d+1}^{i}\binom{k-1}{d-1} = \binom{d-1}{d} + \binom{d-1}{d-1} + \sum_{k=d+1}^{i}\binom{k-1}{d-1} \\ &= \binom{d}{d} + \binom{d}{d-1} + \sum_{k=d+2}^{i}\binom{k-1}{d-1} = \binom{d+1}{d} + \binom{d+1}{d-1} + \sum_{k=d+3}^{i}\binom{k-1}{d-1} = \dots \\ &= \binom{i-3}{d} + \binom{i-3}{d-1} + \sum_{k=i-1}^{i}\binom{k-1}{d-1} = \binom{i-2}{d} + \binom{i-2}{d-1} + \sum_{k=i}^{i}\binom{k-1}{d-1} = \binom{i-1}{d} + \binom{i-1}{d-1} = \binom{i}{d} \end{aligned} \tag{17}$$

The iterative simplification in (17) makes use of the recurrence for Pascal's triangle ([Comtet 1974] [5e]).

Again recalling that each coordinate is restricted to a value no greater than $(j\text{-}1)/2$, let us verify (16) and (17) by way of arguments which, unlike the preceding derivation, avoid generating functions and binomial identities. For $B_j^{C+}(d, i)$, label i tally marks with the integers from 1 to i. Tag each of d tallies, in ascending order of tallies. Tagging the q^{th} tally with the k^{th} tag signifies that the value of the k^{th} coordinate equals the number of tallies after $(k\text{-}1)^{\text{st}}$ tag, up to, and including, the q^{th} tally. Note that there an implicit tag prior to the first tally, and that this construction assures that all coordinates are positive. For the sum of the coordinates to equal i, we must tag the i^{th} tally. This leaves $B_j^{C+}(d,i) = \binom{i-1}{d-1}$ ways to distribute d-1 indistinguishable tags among i-1 distinguishable tallies. Since $V_j^{C+}(d,i)$ corresponds to the case where the sum of the d coordinates is at most i, we are no longer required to tag the i^{th} tally. There are $\binom{i}{d}$ ways to distribute the d tags among the i tallies, and this is the number of positive integer vertices in a d-dimensional ball of L_1 radius i centered at the origin.

Write $V_j^{C\pm}(d, i)$ and $B_j^{C\pm}(d, i)$ for the number of vertices in *resp.* on a d-dimensional ball of L_1 radius i centered at the origin, such that no coordinate is zero. The number of ways of ordering d signs (plus or minus) equals 2^d; each ordering corresponds to a 2^d-tant in d-dimensional space. In consequence,

$$V_j^{C\pm}(d,i) = 2^d\binom{i}{d} \qquad B_j^{C\pm}(d,i) = 2^d\binom{i-1}{d-1} \tag{18}$$

For any k coordinates set to zero, we have $2^{d-k}\binom{i}{d-k}$ *resp.* $2^{d-k}\binom{i-1}{d-1-k}$ vertices in or on a d-dimensional ball of L_1 radius i centered at the origin. Since there are $\binom{d}{k}$ ways of setting k coordinates to zero, the volume is given by

$$V_j^C(d,i) = \sum_{k=0}^{d} 2^{d-k}\binom{d}{k}\binom{i}{d-k} = \sum_{k=0}^{d} 2^{k}\binom{d}{k}\binom{i}{k} = \sum_{k=0}^{i} 2^{k}\binom{d}{k}\binom{i}{k} \qquad (19)$$

The righthand side of (19) explicates how is $V_j^C(d, i)$ is symmetric with respect to i and d. This is in accordance with boundary conditions (9) and (11), recurrence (13), and Table 8, but is to be contrasted with the asymmetric solution to (18):

$$B_j^C(d,i) = \sum_{k=0}^{d} 2^{d-k}\binom{d}{k}\binom{i-1}{d-1-k} = \sum_{k=1}^{d} 2^{k}\binom{d}{k}\binom{i-1}{k-1} = \sum_{k=1}^{i} 2^{k}\binom{d}{k}\binom{i-1}{k-1} \qquad (20)$$

When the radius i exceeds $(j\text{-}1)/2$, a ball centered at the origin of C_j^d (translated) no longer includes all of the points encompassed by the analogous ball (of identical L_1 radius i) in the d-dimensional space of points whose coordinates are integers. For j odd, the ball of interest in C_j^d excludes those points having a coordinate whose absolute value exceeds $(j\text{-}1)/2$; analogous to (15), the ordinary generating function is

$$\left(\frac{x}{1-x}\right)^d\left(1-x^{\frac{j-1}{2}}\right)^d = \sum_{k=1}^{\infty}\binom{d+k-1}{k}x^{d+k}\sum_{q=1}^{d}\binom{d}{q}(-1)^q x^{\frac{(j-1)q}{2}} \qquad (21)$$

wherein for $B_j^{C+}(d,i)$ we extract the coefficient of x^i. Though somewhat more complicated, the case for j even is essentially similar. Rather than pursue this line, we focus on enumerating those points of interest: *i.e.*, those most distant, or most nearly distant, from any given vertex in C_j^d.

Consider points at maximum distance from the origin in an (untranslated) C_j^d, where j is even. Vertex v is maximally distant from the origin if and only if each of the terms in (7) equals $j/2$. This is possible if and only if each coordinate of v equals $j/2$. Thus $(j/2, \ldots, j/2)$ is the unique point at maximum distance $dj/2$ from the origin:

$$B_j^C(d, dj/2) = 1 \qquad j \text{ even} \qquad (22)$$

Again for the case of j even, vertex v is distance $(dj/2)\text{-}1$ from the origin if and only if and only if d-1 terms in (7) equal $j/2$, and one term equals $(j/2)\text{-}1$. The coordinate corresponding to the term whose value equals $(j/2)\text{-}1$ has two possible values: $(j/2)\text{-}1$ and $(j/2)+1$. There are d ways of choosing this term, in which case the remaining d-1 terms are determined. Thus the points at distance one less than the maximum from the origin are those having d-1 coordinates equal to $j/2$ and one coordinate equal to $(j/2) \pm 1$:

$$B_j^C(d, [dj/2]\text{-}1) = 2d \qquad j \text{ even} \qquad (23)$$

Suppose that j is odd. Vertex v is maximally distant from the origin if and only if each of the terms in (7) equals $(j\text{-}1)/2$. Thus the points at maximum distance $d(j\text{-}1)/2$ from the origin have coordinates of the form $((j\pm1)/2, \ldots, (j\pm1)/2)$. That is:

$$B_j^C(d, d(j\text{-}1)/2) = 2^d \qquad j \text{ odd} \qquad (24)$$

Let us apply the notion of opposite pairs to the case of C-cubes: u and v are *opposite* if their distance equals the diameter (alternatively, the radius) $d\cdot\lfloor j/2 \rfloor$ of C_j^d. Vertices u and v are *nearly opposite* if their distance is $d\cdot\lfloor j/2 \rfloor$ - 1, one less than the diameter (alternatively, the radius) of C_j^d.

Theorem 2. Let H be any quorum induced by deleting i vertices from C_j^d, $0 \le i \le f = 2d\text{-}1$, $j \ge 5$. The diameter of H is at least $d\cdot\lfloor j/2 \rfloor$.

Proof. Suppose j is even. By (22), any given vertex u belongs to one opposite pair. Summing over all j^d vertices counts every pair of opposites twice, and the total number of opposite pairs equals $½{\cdot}j^d$. Each vertex we delete from C_j^d removes at most one opposite pair. Therefore, there remains at least one opposite pair as long as

$$4d \le j^d \qquad (25)$$

which follows by noting that $d \le 2^{d-1} \le 5^{d-1} \le j^{d-1}$ Suppose that j is odd. By (22), any given vertex u belongs to 2^d opposite pairs. Summing over all j^d vertices counts every pair of opposites twice, and the total number of opposite pairs equals $2^{d-1}j^d$. Each vertex we delete from C_j^d removes at most 2^d opposite pairs. Therefore, there remains at least one opposite pair as long as

$$(2d\text{-}1)\cdot 2^d < 2^{d-1}\cdot j^d \qquad (26)$$

which reduces to (25). ❒

Theorem 3. Let H be any quorum induced by deleting i vertices from C_j^d, $0 \le i \le f = 2d\text{-}1$, $j \ge 5$. If $i = 0$ or j is odd then the radius of H is at least $d(j\text{-}1)/2$. For $i \ge 1$ and j even, the radius of H is at least $(dj/2)\text{-}1$.

Proof. The case $i = 0$ is covered by Corollary 1.1. Suppose that j is odd. By (24), undeleted vertex u has at least one opposite as long as

$$2d\text{-}1 < 2^d \qquad (27)$$

which follows by inspection. Suppose that j is even. By (23), there is at least one vertex nearly opposite to undeleted vertex u as long as

$$2d\text{-}1 < 2d \qquad (28)$$

which follows since zero is less than one. ❒

path length →	1	2	3	4	5	6	7	8	9	10
000	001	002	003	013	023	033	133	233	333	
000	010	020	030	130	230	330	331	332	333	
000	100	200	300	301	302	303	313	323	333	
000	006	005	004	014	024	034	134	234	334	333
000	060	050	040	140	240	340	341	342	343	333
000	600	500	400	401	402	403	413	423	433	333

path $\rightarrow h$ \ stage $\rightarrow m$			
	0	1	2
	1	2	0
	2	0	1

permutation matrix, cyclic group of order 3

Table 9: Illustration of Theorem 4: $2d = 6$ paths from the origin (0,0,0) to opposite (3,3,3) vertex in a three-dimensional 7-ary C-cube. Swingback paths are listed in the bottom three rows.

Theorem 4. (C-cube connectivity, upper bound on diameter, $j \ge 5$.) If v lies at distance $i > 0$ from vertex u of C_j^d then between u and v there is a set of $2d$ interior-disjoint paths. Let q be the number of coordinates where u and v are identical. i) $d\text{-}q$ of these paths $P(0) \ldots P(d\text{-}q\text{-}1)$ have length i; ii) $2q$ of these paths $P(d\text{-}q) \ldots P(d\text{+}q\text{-}1)$ have length $i\text{+}2$. For $0 \le r \le d\text{-}q\text{-}1$, let c_r^+ denote the value of $\max(|u_k - v_k|, j - |u_k - v_k|)$ that is no larger than any set of $d\text{-}1\text{-}r$ other such c^+'s, (*cf.* (29)) with the ordering ranging over $0 \le k \le d\text{-}1$. iii) Of the remaining $d\text{-}q$ paths $P(d\text{+}q) \ldots P(2d\text{-}1)$, path $P(d\text{+}q\text{+}r)$ traverses $i\text{+}2c_r^+\text{-}j$ edges.

Proof. By induction on d. As a basis take $d = 1$. Since C_j^d is vertex symmetric we can, without loss of generality, suppose that $u_0 = 0$ and $v_0 = i$. For property (i), trace from u to v a path $P(0)$ of minimum length i by traversing i edges along the cycle. Property (ii) holds since q is necessarily 0. For (iii), trace from u to v a path $P(1)$ in a direction opposite to, and interior-disjoint with, $P(0)$; note that $c_0^+ = j\text{-}i$, and that $P(1\text{+}0\text{-}1\text{+}1) = P(1)$ has length $j\text{-}i = i\text{+}2j\text{-}2i\text{-}j = i\text{+}2c_0^+\text{-}j$. The theorem holds at $d = 1$.

Assume that the theorem holds in 0, … $(d\text{-}1)$ dimensions, and regard arbitrary vertices u and v in C_j^d, $d > 1$, $j \geq 4$. Suppose that $q = 0$; *i.e.*, the coordinates of u and v differ in all d dimensions.

i) For the 0th coordinate, trace a shortest path $P'(0)$, of length $\min(|u_0\text{-}v_0|, j\text{-}|u_0\text{-}v_0|)$, from u to $(u_{d\text{-}1}, \ldots, v_0)$. By induction, the $C_j^{d\text{-}1}$ prescribed by setting the 0th coordinate to v_0 contains a path $P''(0)$ from $(u_{d\text{-}1}, \ldots, v_0)$ to v, and this path traverses $i\text{-}\min(|u_0\text{-}v_0|, j\text{-}|u_0\text{-}v_0|)$ edges. Catenating $P'(0)$ with $P''(0)$ gives an i-edge path $P(0)$ from u to v. For $h = 1, \ldots d\text{-}1$, iterate this process to synthesize path $P(h)$: at the start of the hth iteration rotate each coordinate value by adding it to h, and converting the sum to its principal value mod d. As illustrated by the righthand side of Table 9, this completes a symmetric permutation matrix for the cyclic group of order d [Artin 1975] (VII:1.4). At $h = 2$, for example, coordinates along the path change in the order 2, …, d-1, 0, 1. With respect to any vertex along a path, define the *stage* to be the number m of different coordinates that have changed; $q = 0$ implies $0 \leq m \leq d\text{-}1$. Entry (h, m) of the permutation matrix equals $(h+m) \bmod d$. Consider any two paths $P(h_1)$ and $P(h_2)$, for any stage $m < d\text{-}1$. Since entries 0 through m of any row map to successive elements of the cyclic group of order d, at least one of the values in columns 0 through m of row h_1 (*resp.* h_2) must not be in columns 0 through m of row h_2 (*resp.* h_2). But this means that, through stage m, the set of coordinates of $P(h_1)$ that are unchanged from their original values in u differ from the coordinates of $P(h_2)$ that are unchanged from their original values in u. Thus, the only possible intersection of $P(h_1)$ and $P(h_2)$ is at stage d-1. But this is also impossible: the $(h_1 + d\text{-}1 \bmod d\text{-}1)$th coordinate in $P(h_1)$ increments, in a monotone fashion modulo d-1, toward the coordinate value of v in that dimension, while the remaining paths have already attained the coordinate value of v in that dimension. Therefore, any path so constructed is interior-disjoint with any other.

iii) Continuing the case for $q = 0$, construct an additional d paths by substituting a *swingback* at the 0th stage of the preceding procedure. For stages 0 through d-1, that is, begin by tracing a path $P'(d\text{-}1+h)$ of length $\underline{\max}(|u_h\text{-}v_h|, j\text{-}|u_h\text{-}v_h|)$ from u to $(u_{d\text{-}1}, \ldots, v_h{\pm}1 \bmod j, \ldots, u_0)$; if $\max(|u_h\text{-}v_h|, j\text{-}|u_h\text{-}v_h|) = j\text{-}|u_h\text{-}v_h|$ then the zeroth stage path stops at $v_h + 1 \bmod j$; otherwise it stops at $v_h - 1 \bmod j$.

This construction results in a swingback path $P(h)$ passing through a neighbor of v, with the hth coordinate equal to $v_h{\pm}1$. As illustrated by the bottom three rows of Table 9, the final step in the path traverses an edge to v. Note that the total length of $P(h)$ is $i+j\text{-}2|u_h\text{-}v_h|$ if $\min(|u_h\text{-}v_h|, j\text{-}|u_h\text{-}v_h|) = |u_h\text{-}v_h|$; otherwise, $\min(|u_h\text{-}v_h|, j\text{-}|u_h\text{-}v_h|) = j\text{-}|u_h\text{-}v_h|$ and the path length is $i\text{-}j+2|u_h\text{-}v_h|$. In any case, sorting the swingback paths by their lengths yields a set of $d\text{-}q = d\text{-}0 = d$ paths $P(d) \ldots P(2d\text{-}1)$, with $P(d+r)$ traversing $i+2c_r^{+}\text{-}j$ edges, and $0 \leq r \leq d\text{-}q\text{-}1 = d\text{-}1$.

By an argument similar to that pertaining to paths without swingback, any path with swingback intersects no other path (with or without swingback), at least up to the next-to-last edge in the path. As remarked previously, the next-to-last edge advances to a unique neighbor of v (*i.e.*, one which has not been traversed by any other path, with or without swingback). For $q = 0$, that is, any two paths constructed in steps (i) or (iii) are interior-disjoint.

Now suppose that the integer q is positive. With u as source and v as destination, inductively apply the preceding procedure for $q = 0$ to the $d\text{-}q$ coordinates not shared by u and v. i) The $C_j^{d\text{-}1}$ prescribed by the q coordinates whose values are the same in u and v contains $2(d\text{-}q)$ pairwise interior-disjoint u-v paths, $d\text{-}q$ of which traverse i edges.

ii) Construct $2q$ *bypass* paths as follows. If k is the index of a coordinate such that $u_k = v_k$, then traverse to a neighbor of u by crossing one edge in the kth dimension; *i.e.*, by incrementing or decrementing u_k. From this neighbor construct a path to the neighbor of v obtained by incrementing *resp.* decrementing the kth coordinate of v. From u's neighbor to v's neighbor, a single path of length i is guaranteed by applying the procedure for $q = 0$ to the $d\text{-}q$ coordinates not shared by u and v. Traversing from v's neighbor to v completes a path of length i+2. For each such k we obtain two paths (one by incrementing u_k and the other by decrementing u_k), with the k^{th} coordinate unique for every path so constructed. As a result, any bypass path is interior-disjoint with any other bypass path, as well as with any of the 2(d-q) paths (with or without swingback) whose vertex labels vary only in the coordinates not shared by u and v. The bypass procedure constructs 2q paths $P(d\text{-}q)$ … $P(d+q\text{-}1)$ between u and v; each bypass path traverses i+2 edges.

iii) By induction, the $C_j^{d\text{-}1}$ prescribed by the q coordinates whose values are the same in u and v contains d-q paths $P(d+q)$ … $P(2d\text{-}1)$, pairwise interior-disjoint among themselves as well as with those constructed in steps (i) and (ii). Path $P(d+q+r)$ traverses $i+2c_r^{+}\text{-}j$ edges. The theorem holds for $d > 1$. ❐

Corollary 4.1. C_j^d is 2d-connected, and guarantees a quorum in the presence of any 2d-1 faults.

Number *i* of vertices deleted, $0 \le i \le f$ $f = 2\cdot[\log_j n] - 1$	**Radius**		**Diameter**	
	At least	**At most**	**At least**	**At most**
0	$\lfloor j/2 \rfloor \cdot \log_j n$ Corollary 1.1			
from 1 to $[\log_j n]$ - 1	if j is odd then $½\cdot(j\text{-}1)\cdot\log_j n$ else $½\cdot j\cdot[\log_j n]\text{-}1$ Theorem 3	$\lfloor j/2 \rfloor \cdot \log_j n$ Theorem 2, Corollary 4.2		
from $[\log_j n]$ to $2\cdot[\log_j n]$ -2		$\lfloor j/2 \rfloor\cdot([\log_j n]\text{-}1)+\lceil j/2 \rceil$ Corollary 4.2	$\lfloor j/2 \rfloor \cdot \log_j n$ Theorem 2	$\lfloor j/2 \rfloor\cdot([\log_j n]\text{-}1)+\lceil j/2 \rceil$ Corollary 4.2
$2\cdot[\log_j n]$ -1		$\lfloor j/2 \rfloor\cdot(\log_j n)+\lceil j/2 \rceil-1$ Corollary 4.2		$\lfloor j/2 \rfloor\cdot(\log_j n)+\lceil j/2 \rceil-1$ Corollary 4.2

Table 10: Detailed properties of quorums induced by deleting vertices from C-cubes C_j^d, $j \ge 5$, $d \ge 2$.

Let us use our results to formulate upper bounds on quorum diameter at i = 0, 1, … 2d-1 = f faults. Since i = 0 is covered by Corollary 1.1, we focus on $1 \le i \le d\text{-}1$. Although q may assume any value in the range 0 to d-1, the distances of Theorem 3 attain a maximum only if q = 0; *i.e.*, for paths constructed according to procedure (i). To see this, and without loss of generality, note that any two opposites attain the diameter $d\cdot\lfloor j/2 \rfloor$ with i = 0. By contrast, the source and destination of a type (ii) bypass path must be identical in at least one of the coordinates. Therefore, any path constructed according to procedure (ii) has length at most $(d\text{-}1)\cdot\lfloor j/2 \rfloor + 2 \le d\cdot\lfloor j/2 \rfloor$, where the latter follows since $j \ge 5$. For values $1 \le i \le d\text{-}1$, where paths of type (i) or type (ii) apply, it is the type (i) paths which realize the greatest number $d\cdot\lfloor j/2 \rfloor$ of edges.

For a number i of faults in the range $d \le i \le 2d\text{-}1$, consider the length of paths constructed by procedure (iii), with q = 0. For $0 \le r \le d\text{-}q\text{-}1$, define c_r as the value $j - c_r^{+}$; that is, c_r is the (r+1)st greatest addend in <u, v>, the distance (7). Since $c_0^{+} \le \ldots \le c_r^{+} \le \ldots \le c_{d\text{-}1}^{+}$, it follows that

$$c_0 \ge \ldots \ge c_r \ge \ldots \ge c_{d\text{-}1} \tag{29}$$

Writing <P> for the length of path P, express the length of the paths constructed by step (iii) as:

$$<P(d+r)> \;=\; \sum_{k=0}^{r-1} c_k \;+\; (j-c_r) \;+\; \sum_{k=r+1}^{d-1} c_k \tag{30}$$

Consistent with (29), and by the remark preceding Corollary 1.1, the righthand side of (30) is at most

$$rc_0 + (j - c_r) + (d - r - 1)c_{r+1} \;\le\; r\lfloor j/2 \rfloor + (j - c_r) + (d - r - 1)c_{r+1} \tag{31}$$

If $r < d - 1$ then the righthand side of (31) is bounded from above by

$$r\lfloor j/2 \rfloor + (d - r - 2)c_r \;\le\; (d - 1)\lfloor j/2 \rfloor + \lceil j/2 \rceil \tag{32}$$

If $r = d - 1$ then the righthand side of (31) is at most $\quad (d-1)\lfloor j/2 \rfloor + j - c_{d-1} \ \le\ (d-1)\lfloor j/2 \rfloor + j - 1 \qquad (33)$

To complete our analysis of these paths, note that the righthand side of (32) is achieved between any vertices u and v, all d of whose coordinates differ by an absolute value of $\lfloor j/2 \rfloor$ or j-$\lfloor j/2 \rfloor$; for illustration: $u = \mathbf{0}$, $v = (\lfloor j/2 \rfloor, \ldots, \lfloor j/2 \rfloor)$. Further, the righthand side of (33) is achieved between any vertices u and v, d-1 of whose coordinates differ by an absolute value of $\lfloor j/2 \rfloor$ or j-$\lfloor j/2 \rfloor$, and one of whose coordinates differs by ± 1; for illustration: $u = \mathbf{0}$, $v = (\lfloor j/2 \rfloor, \ldots, \lfloor j/2 \rfloor, 1)$. Furthermore, and by remarks following Corollary 4.1, these pathlengths exceed those of paths constructed by procedure (ii). In summary:

Corollary 4.2. Let H be any quorum induced by deleting i vertices from C_j^d, $0 \le i \le f = 2d$-1, $j \ge 5$, $d \ge 2$. If $i \le d$-1 then the diameter of H is at most $d \cdot \lfloor j/2 \rfloor$. If $d \le i \le 2d$-2 then the diameter of H is at most $(d-1)\lfloor j/2 \rfloor + \lceil j/2 \rceil$. If $i = 2d - 1$ then the diameter of H is at most $d \cdot \lfloor j/2 \rfloor + \lceil j/2 \rceil - 1$.

4.4 High Fidelity Results for Quorum Radius and Diameter

The classes $\mathcal{G}$ listed in Table 7 are characterized in terms of the maximum quorum radius ρ, as taken over all fault patterns on $i =$ 0, 1, … f faults. We frame a more accurate picture of performability (equivalently, of performance degradation) by explicating the radius or diameter for each such i. Table 10 frames such a picture in the case of C-cubes, with i increasing as we move down the lefthand column. The solid red curves of Figure 21 plot decreasing performability (*i.e.*, increasing latency, as measured by radius) for a 6-dimensional 5-ary C-cube. This section tabulates results analogous to those of Table 10 for: K-cubes (Table 11), K-cube-connected cycles (Tables 12 and 13), K-cube-connected edges (Table 14), and stars, cycles, and cliques (Table 15). These results lay a solid foundation for performability, a foundation that is readily built upon simulations carried out with STAArchitecture.[62]

Radix j of K-cube	**Number i of vertices deleted, $0 \le i \le f$, $f = [(j\text{-}1) \cdot \log_j n]$ -1**	**Radius**		**Diameter**		**Number i of vertices deleted, $0 \le i \le f$, $f = [(j\text{-}1) \cdot \log_j n]$ -1**
		At least	**At most**	**At least**	**At most**	
2	0	$\log_2 n$ Theorem 7		$\log_2 n$ Theorems 9, 10		from 0 to $[\log_2 n]$ - 2
	from 1 to $[\log_2 n]$- 2	$[\log_2 n]$ - 1 Theorem 11	$[\log_2 n]$ Theorem 9			
	$[\log_2 n]$ - 1		$[\log_2 n]$ + 1 Theorem 9	$\log_2 n$ Theorem 10	$[\log_2 n]$ + 1 Theorem 9	$[\log_2 n]$ - 1
≥ 3	from 0 to $[\log_j n]$ -1	$\log_j n$ Theorems 8, 11		$\log_j n$ Theorems 8, 10		from 0 to $[\log_j n]$ - 1
	from $[\log_j n]$ to $[(j\text{-}1) \cdot \log_j n]$ - 1	$\log_j n$ Theorem 11	$[\log_j n]$ + 1 Theorem 8	$\log_j n$ Theorem 10	$[\log_j n]$ + 1 Theorem 8	from $[\log_j n]$ to $[(j\text{-}1) \cdot \log_j n]$ - 1

Table 11: Detailed properties of quorums induced by deleting vertices of d-dimensional j-ary K-cubes K_j^d. K_j^d is constructible if and only if the maximum number of faults f equals $[(j\text{-}1) \cdot \log_j n]$ -1 and $d = \log_j n$.

We augment Sections 4.1 through 4.3 with sufficient development to enable interpretation of Tables 11 through 15. Let us begin by reviewing constraints on the constructibility of K-cubes. By step (i) on

62. The exposition of Sections 4.3 and 4.5 represents fresh mathematics obtained under, and for, this Phase I NIAC effort. Though *applied* to our STAArchitecture software, results tabulated in Section 4.4 (citations in blue from [LaForge 1999 JPL D-16485]) were derived as part of research sponsored by a 1998 NASA/ASEE Fellowship.

page 34, K_j^d contains j copies of $K_j^{d\text{-}1}$; therefore the order $n_K(d, j)$ of K_j^d equals $j{\cdot}n_K(d\text{-}1, j)$. Subject to the initial condition $n_K(0, j) = 1$, verify that the unique solution of this recurrence relation is $n_K(d, j) = j^{\,d}$ (34)

By step (ii) on page 34, the degree of a vertex in K_j^d equals its degree in $K_j^{d\text{-}1}$ plus j-1, the number of edges that connect it to vertices with the same labels in the other copies of $K_j^{d\text{-}1}$. Subject to the initial condition of zero edges in K_j^0, the degree of each vertex in K_j^d is therefore $d(j\text{-}1)$ (35)

Underlying K-cube $K_j^d(n)$	**Number i of vertices deleted**	**Diameter, as a function of the number i of vertices deleted, $0 \le i \le f = 1 + (j\text{-}1){\cdot}\lfloor \log_j (n/m)\rfloor$**	
		At least	**At most**
radix $j = 2$, dimension $d = \log_2 (n/m) = 1$ number of vertices $n = 2m+1$ odd	0	$1 + \lfloor m/2 \rfloor$, equality by Equation (22), Theorem 15	
	1	$1 + \lfloor m/2 \rfloor$ Theorem 15	$2 + \lfloor m/2 \rfloor$ Theorem 14
	2		$m+1$ Theorem 14
radix $j = 2$, dimension $d = \log_2 (n/m)$ number of vertices $n = m{\cdot}2^d$	0	$\lfloor m/2 \rfloor + \log_2 (n/m)$, equality by Equation (22)	
	from 1 to $[\log_2 (n/m)]$ - 1	$\lfloor m/2 \rfloor + \log_2 (n/m)$ Theorem 15	$\max(2, \lfloor m/2 \rfloor) + \max[2, \log_2 (n/m)]$ Theorem 13
	$\log_2 (n/m)$		$1 + \lfloor m/2 \rfloor + \log_2 (n/m)$ Theorem 13
	$1 + \log_2 (n/m)$		$m - 1 + \log_2 (n/m)$ Theorem 13
radix $j \ge 3$, dimension $d = \log_j (n/m)$ number of vertices $n = m{\cdot}j^d$	0	$\lfloor m/2 \rfloor + \log_j (n/m)$, equality by Equation (22)	
	from 1 to $\log_j (n/m)$	$\lfloor m/2 \rfloor + \log_j (n/m)$ Theorem 15	$\max(2, \lfloor m/2 \rfloor) + \max[2, \log_j (n/m)]$ Theorem 12
	from $1 + \log_j (n/m)$ to $(j\text{-}1){\cdot}\log_j (n/m)$		$1 + \lfloor m/2 \rfloor + \log_j (n/m)$ Theorem 12
	$1 + (j\text{-}1){\cdot}\log_j (n/m)$		$m - 1 + \log_j (n/m)$ Theorem 12

Table 12: Diameter of quorums induced from d-dimensional j-ary K-cube-connected cycles $K_j^d(n)$.

Summing (35) over all $j^{\,d}$ vertices counts every edge twice. Hence the size $e_K(d, j)$ of K_j^d is

$$e_K(d, j) = \tfrac{1}{2}{\cdot}d(j\text{-}1){\cdot}j^{\,d} \qquad (36)$$

As a function of the number of faults present, Table 11 delineates bounds on the quorum radius and diameter of quorums induced from K-cubes. Given the asymptotic optimality of K-cubes (Table 7, Sections 4.2, 4.3, and 4.5), we would like to be able to furnish the designer a wider range of n and f, while maintaining the essential benefits. K-cube-connected cycles and edges serve this purpose.

A *d-dimensional j-ary K-cube-connected cycle of order n*, denoted $K_j^d(n)$, is the result of replacing each of the $j^{\,d}$ vertices of K_j^d with n mod $j^{\,d}$ cycles, each of which contains $\lceil n/j^{\,d} \rceil$ vertices, along with $j^{\,d}$ - n mod $j^{\,d}$ cycles, each of which contains $\lfloor n/j^{\,d} \rfloor$ vertices. Refer to Figure 28. For a basis, a zero-dimen-

Underlying K-cube $K_j^d(n)$	**Number *i* of vertices deleted**	**Radius, as a function of the number *i* of vertices deleted, $0 \le i \le f = 1 + (j\text{-}1)\cdot\lfloor \log_j (n/m) \rfloor$**		**Number *i* of vertices deleted**
		At least	**At most**	
radix $j = 2$, dimension $d = 1 = \log_2 (n/m)$ number of vertices $n = 2m$ even	0, 1 (if m is odd)	$1 + \lfloor m/2 \rfloor$ equality by Equation (22), Theorems 17 and 21		0, 1 (if m is odd)
	1 (if m is even), 2	$\lfloor m/2 \rfloor$ Theorem 17	$1 + \lfloor m/2 \rfloor$ Theorem 21	1 (if m even), 2
radix $j = 2$, dimension $d = 1 = \lfloor \log_2 (n/m) \rfloor$ number of vertices $n = 2m+1$ odd	0, 1 (if m is odd)	if $m = 2$ then 1; else $1 + \lfloor m/2 \rfloor$ equality for $m > 2$ by Theorems 18 and 22		0, 1 (if m is odd) 2 (if $m = 2$)
	1 (if m is even), 2	$\lfloor m/2 \rfloor$ Theorem 18	$1 + \lfloor m/2 \rfloor$ Theorem 22	1 (if m even)
			$1 + \lfloor (m+1)/2 \rfloor$ Theorem 22	2 (if $m > 2$)
radix $j = 2$, dimension $d = \log_2 (n/m) \ge 2$ number of vertices $n = m\cdot 2^d$	0, 1 (if m is odd)	$\lfloor m/2 \rfloor + \log_2 (n/m)$ equality by Equation (22), Theorems 17 and 20		0, 1 (if m is odd)
	1 (if m is even), from 2 to $1 + \log_2 (n/m)$	$\lfloor m/2 \rfloor - 1 + \log_2 (n/m)$ Theorem 17	$\lfloor m/2 \rfloor + \log_2 (n/m)$	1 (if m is even), from 2 to $[\log_2 (n/m)] - 2$
			$\max(2, \lfloor m/2 \rfloor) + \log_2 (n/m)$ Theorem 20	$[\log_2 (n/m)] - 1$
			$1 + \lfloor m/2 \rfloor + \log_2 (n/m)$ Theorem 20	$\log_2 (n/m)$, $1 + \log_2 (n/m)$
radix $j \ge 3$ dimension $d = \log_j (n/m)$ number of vertices $n = m\cdot j^d$	from 0 to $[\log_j (n/m)] - 1$	$\lfloor m/2 \rfloor + \log_j (n/m)$ equality by Equation (22), Theorems 16 and 19		from 0 to $[\log_j (n/m)] - 1$
	from $\log_j (n/m)$ to $[(j\text{-}1)\cdot\log_j (n/m)] - 1$	$\lfloor m/2 \rfloor + \log_j (n/m)$ Theorem 16	if $d = \log_j (n/m) = 1$ then $1 + \lfloor m/2 \rfloor$ else $\max(2, \lfloor m/2 \rfloor) + \log_j (n/m)$ Theorems 19 and 27	$\log_j (n/m)$
	$(j\text{-}1)\cdot\log_j (n/m)$, $1 + (j\text{-}1)\cdot\log_j (n/m)$ (m odd)		if $d = \log_j (n/m) = 1$ then $1 + \lfloor m/2 \rfloor$ else $1 + \lfloor m/2 \rfloor + \log_j (n/m)$ Theorems 19 and 27	from $1 + \log_j (n/m)$ to $1 + (j\text{-}1)\cdot\log_j (n/m)$
	$(j\text{-}1)\cdot\log_j (n/m)$, $1 + (j\text{-}1)\cdot\log_j (n/m)$ (m even)	$\lfloor m/2 \rfloor - 1 + \log_j (n/m)$ Theorem 16		

Table 13: Radius of quorums induced from d-dimensional j-ary K-cube-connected cycles $K_j^d(n)$.

sional K-cube-connected cycle $K_j^0(n)$ is a cycle with vertices labeled from 0 to $\lfloor n/j^0 \rfloor - 1$ (*i.e.*, from 0 to n-1). The high order d digits of the label on a vertex u in cycle h of $K_j^d(n)$ are identical to the d digits on the label of vertex h of the corresponding K_j^d. The low order digit on u is its label in the corresponding $K_j^0(n)$. Vertex u shares an edge with vertex v if and only if i) u and v are neighbors in a basic cycle $K_j^0(n)$; or ii) the low order digits of u and v are identical, and the high order digits differ in exactly one position, or iii) there are $\lfloor n/j^{\,d} \rfloor$ vertices in the basic cycle of which *u* is a member, $\lceil n/j^{\,d} \rceil$ vertices in the basic cycle of which *v* is a member, and *u* and *v* have the highest labels in their respective basic cycles.

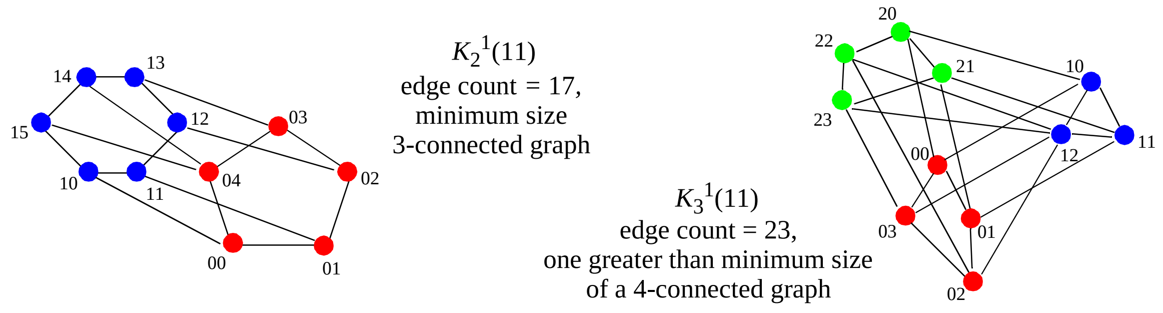


Figure 28: A K-cube-connected cycle $K_j^{\,d}(n)$ has minimum size if and only if (41a) or (41b) holds.

Since each basic cycle must contain at least three vertices, it follows that $\qquad n/3 \geq j^{\,d} \qquad$ (37)

is a constraint on the number of vertices in any $K_j^{\,d}(n)$. If $0 = n \bmod j^{\,d}$ then $n = m{\cdot}j^{\,d}$ for some positive integer *m*. Since it contains exactly *m* vertices per basic cycle, we denote such a $K_j^{\,d}(m{\cdot}j^{\,d})$ by $K_{m\cdot j}^{\,d}$.

Each vertex of $K_{m\cdot j}^{\,d}$ has degree $\qquad d(j\text{-}1)+2 = f+1 \qquad$ (38)

Summing the degree of every vertex counts each edge twice, hence

$$e_K(d, j, n) = ½{\cdot}m{\cdot}j^{\,d}{\cdot}(d[j\text{-}1] + 2) = ½{\cdot}m{\cdot}d{\cdot}j^{\,d}[j\text{-}1]+m{\cdot}j^{\,d} \qquad (\text{for } n = m{\cdot}j^{\,d}) \tag{39}$$

Since either *j* or *j-1* is even, the first term on the righthand side of (39) is an integer; (39) is therefore an integer. Substituting $d(j\text{-}1)+2 = f+1$, we see that (39) equals $\lceil n(f+1)/2 \rceil$. Thus $0 = n \bmod j^{\,d}$ implies that the number of edges in $K_{m\cdot j}^{\,d}$ is exactly that of any minimum size $(f+1)$-connected graph on $m{\cdot}j^{\,d}$ vertices. Suppose on the other hand that $0 \neq n \bmod j^{\,d}$. By step (iii) above, we connect the vertex with the highest label in each of the $n \bmod j^{\,d}$ long cycles to the vertex with the highest label in each of the $j^{\,d} - n \bmod j^{\,d}$ short cycles; moreover, we count these $(n \bmod j^{\,d})(j^{\,d} - n \bmod j^{\,d})$ “extra” edges only at d=1.

Summing the degree of each vertex counts each edge twice. The number of edges in $K_j^{\,d}(n)$ is therefore

$$e_K(d, j, n) = \lceil ½{\cdot}[\, n{\cdot}(d[j\text{-}1]+2) + (n \bmod j^{\,d})(j^{\,d} - n \bmod j^{\,d})] \rceil \tag{40}$$

Substituting $d(j\text{-}1)+2 = f+1$, we see that (40) equals $\lceil ½{\cdot}[\, n(f+1) + (n \bmod j^{\,d})(j^{\,d} - n \bmod j^{\,d})] \rceil$. That is, $K_j^{\,d}(n)$ has minimum size $\lceil n(f+1)/2 \rceil$ if and only if

<u>either</u> $\quad$ a) $\quad 0 = n \bmod j^{\,d} \qquad$ <u>or</u> $\quad$ b) $\quad j = 2,\ d = 1,\ f = 2,\ n$ odd $\qquad$ (41)

By comparison to K-cubes, our K-cube-connected cycles must satisfy *three* constraints: (37), (38), and (41). Despite this, a *(d*-2*)*-dimensional K-cube-connected cycle may be constructible where the corresponding *d*-dimensional K-cube is not. Note that (41) says quite a bit about the structure of K-cube-connected cycles $K_j^{\,d}(n)$ of size $\lceil n(f+1)/2 \rceil$: <u>either</u> $K_j^{\,d}(n)$ is a $K_{m\cdot j}^{\,d}$, <u>or</u>, for all *n* and $f = 2$, $K_j^{\,d}(n) = K_2^{\,1}(n)$ comprises two cycles, one with $\lceil n/2 \rceil$ vertices, the other consisting of $\lfloor n/2 \rfloor$ vertices. The latter holds since if *n* is not odd then 2 divides *n*; in this case (41a) is satisfied, and we have a $K_{m\cdot 2}^{\,1}$. In particular, the size of any one-dimensional binary K-cube-connected cycle is the same as that $\lceil 3n/2 \rceil$ of a 3-connected graph with fewest edges. Note that our definition of a K-cube-connected cycle is somewhat different from that described by [Preparata and Vallemin 1981] and analyzed by [Banerjee et al 1986]. Elsewhere in this report we write m in place of the integer value $\lfloor n/j^d \rfloor$, the least number of vertices in a cycle. As a function of the number of faults present, Tables 12 and 13 delineate bounds on the quorum diameter *resp.* radius of quorums induced from K-cube-connected cycles.

$K_{2\cdot 3}^{1} = K_{3\cdot 2}^{1}$
edge count = 9,
minimum size
3-connected graph
on 6 vertices;
maximum radius of quorum
minimized at 2

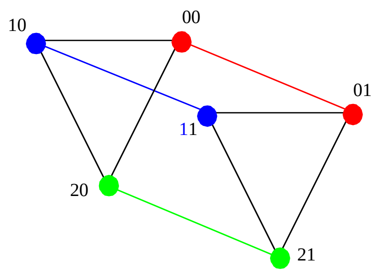


Figure 29: K-cube-connected edge $K_{2\cdot j}^{d}$, radix j = 3. At j = 2, $K_{2\cdot 2}^{d}$ reduces to a binary K-cube K_2^{d+1}.

The structure of a K-cube-connected edge lies between that of K-cubes and K-cube-connected cycles. Refer to Figures 2 and 29. A *d-dimensional j-ary K-cube-connected edge of order n*, denoted $K_{2\cdot j}^{d}$, is the result of replacing each of the j^d vertices of K_j^d with an edge. For a basis, a zero-dimensional K-cube-connected edge $K_{2\cdot j}^{0}$ is an edge connecting two vertices. The high order d digits of the label on a vertex u in edge h of $K_{2\cdot j}^{d}$ are identical to the d digits on the label of vertex h of the corresponding K_j^d. The low order digit on u is its label in the corresponding $K_{2\cdot j}^{0}$. Vertex u shares an edge with vertex v if and only if i) u and v are neighbors in a basic edge $K_{2\cdot j}^{0}$, or ii) the low order digits of u and v are identical, and the high order digits differ in exactly one position. This definition gives rise to a development analogous to that for K-cube-connected edges. For example, the respective counterparts to (37) and (38) are

$$n/2 = j^d \tag{42}$$

and

$$d(j\text{-}1)+1 = f+1 \tag{43}$$

On the other hand, (41) pertains intact. Except for the case n = 5, therefore, a K-cube-connected graph with given connectivity and minimum size cannot have as its basis a mixture of edges and cycles. It is for this reason that we have equality in (42). As a function of the number of faults present, Table 14 delineates bounds on the quorum radius and diameter of quorums induced from K-cube-connected edges.

Number i of vertices deleted, $0 \le i \le f$, $f = (j\text{-}1)\cdot\log_j(n/2)$	**Radius**		**Diameter**	
	At least	**At most**	**At least**	**At most**
from 0 to $\log_j(n/2)$	$1 + \log_j(n/2)$ Theorems 23 and 25		$1 + \log_j(n/2)$ Theorems 23 and 24	
from $1 + \log_j(n/2)$ to $(j\text{-}1)\cdot\log_j(n/2)$	$1 + \log_j(n/2)$ Theorem 23	if $d = \log_j(n/2) = 1$ then 2 else $2 + \log_j(n/2)$ Theorems 23 and 27	$1 + \log_j(n/2)$ Theorem 24	$2 + \log_j(n/2)$ Theorem 23

Table 14: Properties of quorums induced by deleting vertices from K-cube-connected edges $K_{2\cdot j}^{d}$, $j \ge 3$.

To conclude this section, Table 15 details the unique minimum size graphs with minimum quorum radius and diameter, at the minimum (f = 0, 1) and maximum (f = n-2, n-1) endpoints of fault tolerance. We defined a clique K_n on page 33; a *star* S_n is a tree having a single central vertex; a *cycle* C_n is a path whose endpoints are joined by an edge. Table 15 implies the results listed in Table 7, at f = 0, 1, n-2, and n-1.

Fault tolerance f	$\mathcal{G}^+_{n,f,k}$: $(f+1)$-connected graphs of minimum size $\lceil n(f+1)/2 \rceil$, induced quorums have radii at most k	Radius of quorum and of tree spanning quorum, as a function of the number $i \le f$ of vertices deleted	Diameter of tree spanning quorum, as a function of the number $i \le f$ of vertices deleted	References
0	Best possible $\mathcal{G}_{n,0,1}$ uniquely the set of n-vertex stars S_n	1	2	Figure 4 Theorem 3
1	Best possible $\mathcal{G}_{n,1,\lfloor n/2 \rfloor}$ uniquely the set of n-vertex cycles C_n	$\lfloor n/2 \rfloor$ if $i = 0$ $\lceil n/2 \rceil - 1$ otherwise	$n - 1$ if $i = 0$ $n - 2$ otherwise	Figure 5 Theorem 4
$n - 2$	Best possible $\mathcal{G}_{n,n-2,1}$ uniquely the set of n-vertex cliques K_n	1	1 if $i = f$ 2 otherwise	Figure 6 Theorem 5
$n - 1$	Best possible $\mathcal{G}_{n,n-1,1}$ uniquely the set of n-vertex cliques K_n	0 if $i = f$ 1 otherwise	0 if $i = f$ 1 if $i = f - 1$ 2 otherwise	Discussion following Theorem 5

Table 15: Characteristics of quorums at either end of the range of the fault tolerance $f < n$, $n \ge 3$.

4.5 Performability of Large Scale Architectures

Complementing Table 7, Tables 10 through 15 provide a taxonomy that includes, at the minimum and maximum fault tolerance, minimum size graph architectures (stars, cycles, and cliques) with minimum latency. This section addresses the theoretical optimality of K-cubes, K-cube-connected edges, K-cube-connected cycles, and C-cubes. To preview: in a ratioed asymptotic sense, K-cube constructions can deliver the best possible value $\Theta(\log n)$ of $\rho(n, f)$; *i.e.*, a quorum radius that, within a constant factor (perhaps equal to one) matches the lower bounds $\rho^-_{\text{Thm 6}}$ of inequality (1). Moreover, K-cubes and their relatives are preferred to C-cubes for two reasons: 1) the radius of a C-cube quorum exceeds the diameter of the comparable K-cube having identical fault tolerance; 2) there is *no* relation between j and d such that, as $n_C = j^d \to \infty$, the ratio of the C-cube quorum radius to $\rho^-_{\text{Thm 6}}$, does *not* diverge; *i.e.*, this ratio must approach infinity. With respect to both criteria, that is, C-cubes are sub-optimal.

For real x, the sign of x is indicated by the function signum(x). If $x > 0$ then signum$(x) = 1$; if $x < 0$ then signum$(x) = -1$; if $x = 0$ then signum$(x) = 0$. Refer to Table 7. The signum function allows us to conveniently encapsulate the fault tolerance of $K_{m \cdot j}{}^d$ as

$$f = (j-1) \cdot d + \text{signum}(m-2) \tag{44}$$

Theorem 5. Denote by $\rho^-_{\text{Thm 6}}$ the lower bound on the radius of any quorum, as given by inequality (1).

If $\rho^-_{m,j,d} = \log_j (n/m) + \lfloor m/2 \rfloor$ and $\rho^+_{m,j,d} = 1 + \log_j (n/m) + \lfloor m/2 \rfloor$ (45)

are the minimum *resp.* maximum radius of quorums of $K_{m \cdot j}{}^d$, as listed in Table 7, then

$$\frac{\left[d + \left\lfloor \frac{m}{2} \right\rfloor\right][\ln (j-2) + \ln d]}{\ln m + d \ln j} \le \frac{\rho^-_{m,j,d}}{\rho^-_{\text{Thm 6}}} \le \frac{\rho^+_{m,j,d}}{\rho^-_{\text{Thm 6}}} \le \frac{\left[d + \left\lfloor \frac{m}{2} \right\rfloor + 1\right][\ln j + \ln d]}{\ln m + d \ln j - 1.4} \tag{46}$$

Proof. Explicate $\rho^-_{m,j,d}$, $\rho^+_{m,j,d}$, and $\rho^-_{\text{Thm 6}}$, the latter without the ceiling function. Making use of (44), substitute $j = 1 + [f - \text{signum}(m-2)]/d$. For the lower bound invoke the inequalities $-1 \le \text{signum}(m-2)$, $n(f-1)+3 \le nf$, and $\ln[(f-1)/(f+2)] < 0$. For the upper bound observe that $\text{signum}(m-2) \le 1$, $(j-1)d+2 \le jd$, and $-1.4 < \ln[(f-1)/(f+2)]$. The result follows by algebraic manipulation. ❐

It is interesting to note that, in the large, the fault tolerance (44) of $K_{m \cdot j}{}^{d}$ is dominated by *j* and *d*, and grows in a fashion that is independent of *m*. By contrast, the radius of $K_{m \cdot j}{}^{d}$ is dominated by *m* and *d*, and is independent of *j*. Our conclusions about the optimality of the quorum radius of $K_{m \cdot j}{}^{d}$ depend on how *m*, *j*, and *d* tend to infinity. If the left and right sides of (46) tend to some limit λ then, in the large, $\rho^-_{m,j,d}$, $\rho^+_{m,j,d}$, and $\rho^-_{\text{Thm } 6}$ are within a factor λ of $\rho(m \cdot j^{d}, (j\text{-}1) \cdot d + \text{signum}[m\text{-}2])$, the minimum value (over all graphs) of the maximum quorum radius. Abbreviating the latter quantity as $\rho_{m,j,d}$, we obtain the following result.

Corollary 5.1. For all $n_K = m \cdot j^{d} \ge k$, if *q* and *r* are least upper bounds such that $\left\lfloor \frac{m}{2} \right\rfloor + 1 \le qd$ and $\ln d \le r \ln j$, then $\rho^+_{m,j,d} \le (\rho_{m,j,d}) \cdot (1 + q + qr + r)$.

Under the conditions of Corollary 5.1, that is, the maximum quorum radius of $K_{m \cdot j}{}^{d}$ approaches a value that is within a factor 1 + *q* + *qr* + *r* of the minimum. Several special cases of Corollary 5.1 are of particular interest: a) $d \in o(j)$; b) $m \in o(d)$; c) both (a) and (b). In this instance the maximum radius of quorums induced from $K_{m \cdot j}{}^{d}$ is asymptotically within a factor a) $1 + q$, b) $1 + r$, or c) 1 of $\rho_{m,j,d}$.

If both *m* and *d* are bounded then the only way for the number of vertices to approach infinity is for the radix *j* to increase. In this case we can improve Corollary 5.1 to best possible.

Corollary 5.2. If $d, m \in \Theta(1)$ then $\lim_{n_K \to \infty} \frac{\rho^+_{m,j,d}}{\rho_{m,j,d}} = \frac{\rho^-_{m,j,d}}{\rho_{m,j,d}} = 1$.

In the ratioed asymptotic sense of Corollaries 5.1 and 5.2, both the lower bounds $\rho^-_{\text{Thm } 6}$ of inequality (1) and the quorum radius of $K_{m \cdot j}{}^{d}$ are best possible. In other cases it may be that one of these bounds is best possible, but this remains to be proved. We also stress that $\rho^+_{m,j,d} / \rho^-_{\text{Thm } 6}$ and $\rho^-_{m,j,d} / \rho^-_{\text{Thm } 6}$ approach one quite slowly. The reason for this appears to be the ln *j* factors in the expressions of (46). The 125-node K-cube of Figure 21 illustrates a case where $\rho^+_{m,j,d} / \rho^-_{\text{Thm } 6}$ equals 2. As computed by STAArchitecture, for example, at $(n, f) = (121, 19)$ and $(n, f) = (512, 20)$ we have $(m, j, d) = (1, 11, 2)$ and $(m, j, d) = (1, 8, 3)$; the corresponding ratios $\rho^+_{m,j,d} / \rho^-_{\text{Thm } 6}$ are 3/2 and 4/3.

Before presenting the last two theorems of this section, let us review our terminology. Refer to the two middle columns of Table 7. By the *maximum radius* $\rho(n, f)$ we mean the largest radius of any quorum induced by *f* or fewer faults. Thus, for example, to obtain a *lower* bound on the maximum radius of a K-cube (*resp.* C-cube), we take the largest of the lower bounds on radii as listed in Table 11 (*resp.* Table 10); for an *upper* bound on the maximum radius of a K-cube or C-cube quorum, we take the largest of the upper bounds on radii as listed in Table 11 *resp.* Table 10. Similarly, introduce the *maximum diameter* $\Delta(n, f)$ as the largest diameter of any quorum induced by *f* or fewer faults. Thus, for example, to obtain a *lower* bound on the maximum diameter of a K-cube (*resp.* C-cube) quorum, we take the largest of the lower bounds on diameter as listed in Table 11 (*resp.* Table 10); for an *upper* bound on the maximum diameter of a K-cube or C-cube, we take the largest of the upper bounds on diameter as listed in Table 11 *resp.* Table 10. Finally, note that if *f* is the worst-case fault tolerance of an *n*-vertex graph architecture, then the *fractional fault (worst-case) tolerance* is simply $f_{\text{frac}} = f / n$. With these notions in hand, we can quantify relative merit of K-cubes and C-cubes.

Theorem 6. If the worst-case fault tolerance *f* of $K_j{}^{d}$ equals that of $C_J{}^{D}$ then, for $j, J \ge 5$, $d, D \ge 2$:

The maximum diameter Δ_K of $K_j{}^{d}$ is less than the maximum radius ρ_C of $C_J{}^{D}$: $\Delta_K < \rho_C$ (47)

The order $n_K(j,d)$ of $K_j{}^{d}$ is less than the order $n_C(J,D)$ of $C_J{}^{D}$: $n_K < n_C$ (48)

Proof. By hypothesis and Corollary 4.1: $f + 1 = d(j - 1) = 2D$ (49)

By Table 11: $\Delta_K \le d + 1$ (50)

By Table 10, and by inequalities (49) and (50): $½\cdot D(J - 1) = ¼\cdot d(j - 1)(J - 1) \le \rho_C$ (51)

For (47) it therefore suffices to show $d + 1 < ¼\cdot d(j - 1)(J - 1)$ (52)

But (52) holds since $j, J \ge 5$, $d, D \ge 2$, and $1 + 1/d \le 2 < 4 \le j - 1$ (53)

Now note that, for integers $r > q \ge 5$, we have $r/q < 6/5 < 1.7 < 2 < 5^{½}$. Hence $5^{½\cdot(q-1)}/q < 5^{½\cdot(r-1)}/r$ and the value of $j/5^{½\cdot(j-1)}$ decreases strictly with increasing integer $j \ge 5$. In particular, since $5 < 5^2 = 25$, and since $J \ge 5$, $d \ge 2$, we can make use of (49): $n_K = j^d < 5^{½\cdot d(j-1)} \le J^{½\cdot d(j-1)} = J^D = n_C$ (54)

Thus, (47) and (48) hold. ❐

Inequality (47) of Theorem 6 says that, for given fault tolerance, the maximum diameter of K-cube quorums is less than the maximum radius of C-cube quorums. Moreover, (48) establishes that the worst-case *fractional* fault tolerance of K-cubes is superior to that of C-cubes. Theorem 6 focuses on radices greater than 4 and dimensions greater than 1 since, for $j \le 4$ or $d = 1$, C-cubes are isomorphic to K-cubes or K-cube-connected cycles. But in how many cases can the fault tolerance of a C-cube equal that of a K-cube? That is, for what constructions is the degree of each vertex in a K-cube equal to that f+1 of any vertex in a C-cube? By inspection of (49), such a construction is realized if and only the degree of every vertex of the K-cube is an even integer no less than eight. In other words, for $j > 4$ and $d > 1$, Theorem 6 applies to all C-cubes; moreover, Theorem 6 applies to a subset of K-cubes (loosely speaking, "half" of them) that map many-to-one onto the set of C-cubes.

Despite Theorem 6's quantitative preference for K-cubes over C-cubes, it seems plausible that, when divided by $\rho^-_{\text{Thm 6}}$, the maximum radius of C-cube quorums attains a limit, akin to that expressed by Corollaries 5.1 and 5.2. That is, we still do not know whether, for some scaling of j and d, the maximum radius of quorums induced from C_j^d is asymptotically within a constant factor of $\rho^-_{\text{Thm 6}}$. Alas, such scalability is impossible, as the next theorem shows.

Theorem 7. As $n_C(j, d) = j^d$ tends to infinity, the ratio $\rho_C(j^d, 2d\text{-}1)/\rho^-_{\text{Thm 6}}$ grows without bound.

Proof. Suppose to the contrary that $\rho_C/\rho^-_{\text{Thm 6}} \in \Theta(1)$. Then for some j, d, and k corresponding to all $n_C(j, d) \ge k$, the ratio is bounded from above by a least constant $b \ge 1$. As with Theorem 5, we employ simplifying substitutions to consider

$$\frac{(j-1)\ln d}{\ln j} \le 2\left[\frac{\left\lfloor \frac{j}{2} \right\rfloor d \ln d}{d \ln j}\right] = \frac{2\bar{\rho}_C}{\rho^-_{\text{Thm 6}}} \le 2b \qquad (55)$$

for such sufficiently large $n_C(j, d) \ge k$. The scaling condition $n \to \infty$ implies that $(j\text{-}1)\cdot\ln d \to \infty$. Hence, for the upper bound b to exist, the denominator on the lefthand side of (55) must approach infinity: $\ln j \to \infty$. But this means that $j \to \infty$. As $j \to \infty$, $(j\text{-}1)/\ln j$ grows without bound; hence there can be no k such that, for all $n_C(j, d) \ge k$, (55) is satisfied. That is, $\rho_C(j^d, 2d)/\rho^-_{\text{Thm 6}}$ grows without bound. ❐

Theorem 7 says that lower bounds $\rho^-_{\text{Thm 6}}$ of inequality (1) (a variant on the *Moore bound* mentioned on page 33) cannot be achieved by C-cubes, even in the sense of asymptotic ratios. This is *not* the same as a wholesale assertion about the ratio of C-cube quorum radii to the optimum value of the maximum radius $\rho(n, f)$, and we are not in a position to advance such a claim. However, for scaling trends that enable K-cubes to come within a constant factor of $\rho(n, f)$, we *can* be certain that the ratio $\rho_C/\rho(n_C, f)$ diverges. More precisely:

Corollary 7.1. For $(j\text{-}1)d$ even, let j and d be the radix and dimension of the class of K_j^{d} such that $d \in \Theta(1)$ or, with r the least upper bound such that, for all $n_C = j^d \geq k$, $\ln d \leq r \ln j$. Let $\{C_J^{½(j\text{-}1)d = D}\}$ be the class of C-cubes corresponding to such K_j^{d}'s, as prescribed by the discussion following Theorem 6. If $n_K(j, d) = j^d$ tends to infinity then, by equation (48) of Theorem 6, n_C tends to infinity; moreover, by Theorem 7, the ratio $\rho_C / \rho(n_C, f)$ grows without bound.

4.6 STAArchitecture Simulates Distributed Algorithms for Diagnosis and Configuration

If quorums are to self-configure in accordance with criterion 4.1.1 then nodes must have a means of resolving which nodes are healthy. By contrast to configuration, results for mutual test and diagnosis (MTAD, requirement 3.8) are relatively well-established. Comparing Figure 9 to the algorithms of Figure 22, we see than the demarcation between diagnosis and configuration is not necessarily sharp. In part because of its simplicity, the second algorithm of Figure 22 readily generalizes to *any* graph architecture:

```
Generalized Diagnosis and Configuration Algorithm $\mathcal{A}_{\text{connected}}$   % Configure a quorum from any
1)    For each of $u_i$'s neighbors $j$                                               %  graph architecture.
2)          If     test $(u_i, u_j)$ fails                                             %    N.B. runs on $u_i$.
3)          then  $u_i$ disables its port to $u_j$                                     %  Local test is diagnosis
```

The correctness and efficiency of $\mathcal{A}_{\text{connected}}$, indeed any MTAD algorithm, boils down to:

i) What constitutes a test? ii) What is the local test coverage? iii) What are blocking constraints on tests? iv) How does the efficacy of diagnosis and configuration vary with test redundancy and local coverage?

Suppose, for example, we build our system so that (ii) healthy nodes detect low-level faults in their neighbors with test coverage approaching 100%. [Bianchini and Buskens 1992] and [Muraldi 1990] show how we can effectively achieve this using (i) a combination of software and circuitry that exercises the point-to-point connections and operating system or agent on each end. It follows that (iv) the minimum size of an n-node graph architecture that is capable of diagnosing *and* configuring f faults *equals* the size $\lceil n(f+1)/2 \rceil$ of an n-node graph architecture that is capable of forming an n-f node quorum (with diagnosis already performed) in the presence of f faults [Preparata et al 1967]. In the worst case, that is, the optimum redundancy for test and configuration equals the optimum redundancy for configuration alone. Recalling requirement 3.6, if f is a constant fraction p of n then the discrete redundancy of combined diagnosis and configuration is $\lceil ½(n^2p+n) \rceil \in \Theta(n^2)$, *quadratic* in n. Further, if each node has the power of (say) a linear bounded automaton [Hopcroft and Ullman 1979], then we have more than ample resources to assure asynchronous negotiation and execution of tests between each node and its neighbors. In this case the (serialized) efficiency of $\mathcal{A}_{\text{connected}}$ is at most the maximum degree of a node; *i.e.*, $f+1 = np+1 \in O(n)$ for minimum size architectures with a constant proportion of faulty nodes. If (iii) tests are pairwise exclusive (*i.e.*, each node is either being tested, testing, or idle) then the running time of any MTAD algorithm is $\Omega(f+1) \in O(n)$; this bound which is achieved whenever the size of the smallest one-factorization is $O(f+1) \in O(n)$.[63] A similar, holistic approach extends to other models of cost, benefits, and faults.

Let us revisit an issue first touched on in Sections 3.6 and 3.14: *probabilistic* models. A compelling reason to consider *almost sure* diagnosis or configuration is that the cost or benefit (*e.g.*, redundancy, algorithmic complexity, fault tolerance) is usually better than in the worst case – by *orders of magnitude*.[64] Refer to

63. A *one-factorization* is a covering by *matchings*, a special case of coverings by trees. The number of matchings in a one-factor is at least the *arboricity* of the graph. See also footnote 55.

64. One notable exception is the case of arrays spared by rows and columns: here the probabilistic and worst-case fault tolerance have the same order of magnitude [LaForge 1999 Trans. Reliability].

Table 16. By contrast with the worst case, for example, suppose that we insist only that the probability of configuring a quorum be *close* to one (*i.e.*, $1-o(1)$) in the presence of a constant proportion of independent, identically distributed (iid) faults. In this case the price we pay for redundancy (size divided by order) drops from $\Theta(n)$ to $\Theta(\log n)$. Furthermore, and as summarized by the bottom row of Table 16, we can decrease redundancy to a best possible constant $\Theta(1)$, as long as we insist that a quorum only contain *almost* every healthy node. In addition to the features described in Sections 4.1 through 4.5:

4.6.1 STAArchitecture synthesizes self-configuring architectures that
a) tolerate a constant proportion of faults (req. 3.6)
b) maintain connectivity among healthy nodes, and that disconnect healthy from faulty nodes (req. 3.7);
c) identify faults via MTAD, mutual test and diagnosis (req. 3.8)

Criteria for quorum inclusion	**Probability that criteria are met**	**Distribution of *np* faults**	**Behavior of faulty elements**	**Test redundancy**
100% accuracy [Preparata et al 1967] [Hayes 1976]	1	anywhere (worst case) $p < ½$	malicious (worst case)	$\Theta(n)$
100% accuracy [Blough 1988] [Scheinerman 1987]	$1-o(1)$	**iid** $p < ½$	malicious (worst case)	$\Theta(\log n)$
100% of faulty elements; $(1 - \varepsilon)\cdot 100\%$ of healthy elements [LaForge et al 1993]	$1-O(n^{-1})$ arbitrarily small $\varepsilon > 0$	**iid** $0 < p < 1$	malicious if $p < 0.5$ else faulty can agree with probability $p\cdot e^{-0.847/(1-p)}$	$\left\lceil \frac{c + 3.4044}{-\ln p} \right\rceil \in \Theta(1)$ constant c depends on ε

Table 16: Correctness and fault model versus redundancy for self-healing diagnosis and configuration.

As Figures 30 through 32 illustrate, the proportion of faults that can we tolerate is as high as 85%.[65] Building a multicomputer with this level of fault tolerance is both necessary and achievable for starship autonomy. The remainder of this section elaborates conditions under which we know this claim holds. We point out open issues whose resolution will further enable self-healing architectures and algorithms, and further relate these issues to the exposition of Sections 4.1 through 4.5.

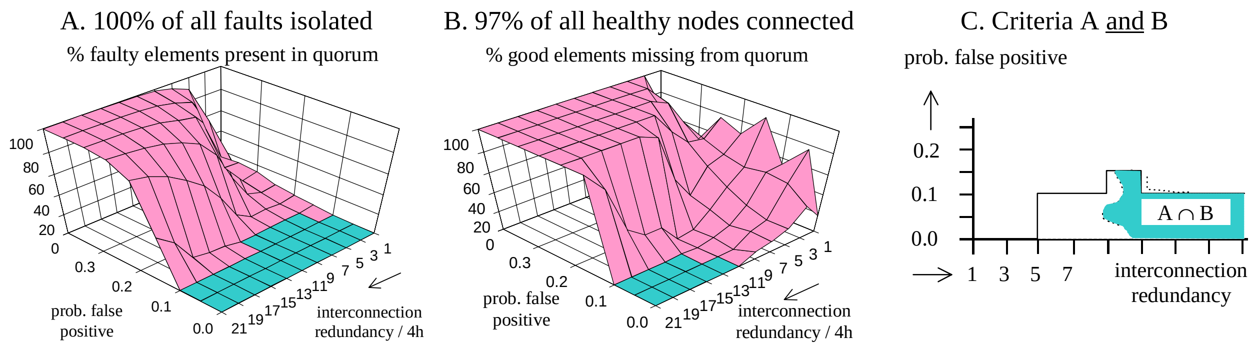


Figure 30: Feasible regions of design for self-healing algorithms and architectures. 85% of nodes faulty. Algorithm $\mathcal{A}_{\text{connected}}$ combined with architecture *G(h;* 5, 5*)*. Boundary chaos remains unexplained.

65. This is considerably better than our Phase I target of 33% ([LaForge 1999 NIAC Phase I Proposal] Table 1)

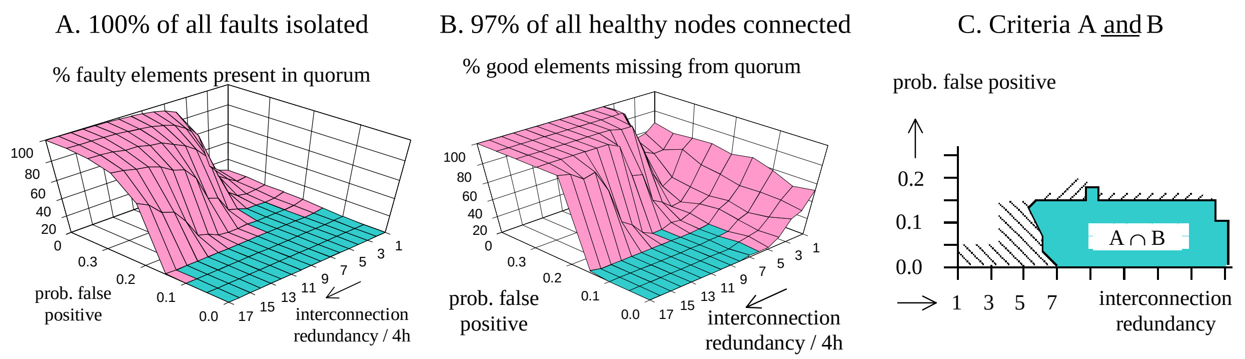


Figure 31: Feasible regions of design for self-healing algorithms and architectures. 75% of nodes faulty. Algorithm $\mathcal{A}_{\text{connected}}$ combined with architecture *G(h;* 5, 5*)*. Compare complexity with Figure 32.

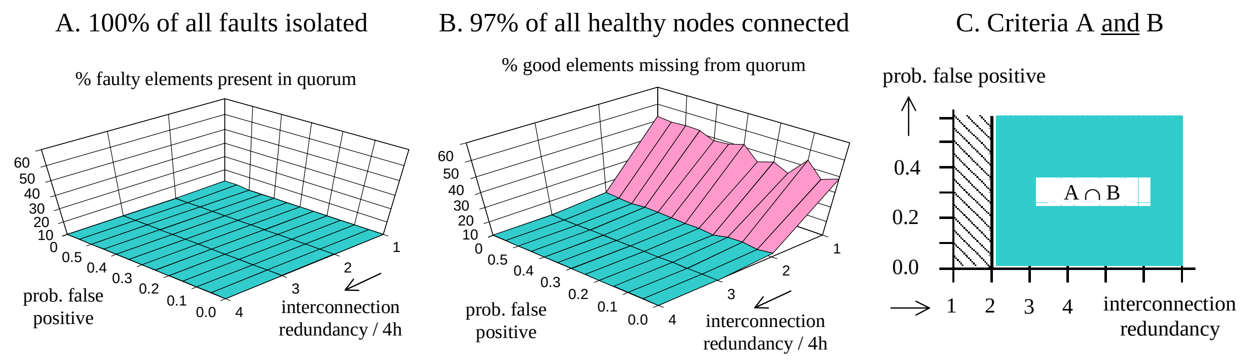


Figure 32: Quorum feasibility: diagnosis combined with configuration, 45% of nodes faulty. Algorithm $\mathcal{A}_{\text{connected}}$ combined with architecture *G(h;* 8, 8*)*. *h* measures the discrete redundancy.

To begin, the righthand column of the worst-case results of Table 16 pertains to a specific class of graph architectures: the regular chordal graphs referenced on page 26 of Section 4.2.[66] Although STAArchitecture can synthesize these graphs for any n and f, the resulting latency is poor (*cf.* discussion, page 27). On the other hand, our d-dimensional K-cube structures have low latency, but exhibit fault tolerance on the order of $j \cdot \log_j n$. Thus, while K-cubes represent an advance for performability (especially when compared to C-cubes), the worst-case fault tolerance of K-cubes falls short of constant fractional fault tolerance:

4.6.2 It remains to characterize architectures satisfying 4.1.1, with $f = np$, in the worst case.

At the other end of the spectrum, the architectures *G(h; s, t)* referenced by Figures 30 through 32 deliver fault tolerance in constant proportion to the number of nodes, and at constant redundancy (in this case, by either count of edges or VLSI layout area [LaForge and Korver 2000 MTAD]). However, the probabilistic radius or diameter of *G* appears to be on the order of the square root of *n*. We do not know how close (or how far!) this latency is from optimal, and it remains to garner sharp results along these lines:

4.6.3 It remains to characterize architectures satisfying 4.1.1, with $f = np$, in the *probabilistic* case.

Finally, a caveat about the assumption that healthy nodes apply tests with 100% test coverage. Preliminary simulations indicate that 4.6.1 remains valid when this condition is relaxed (*e.g.*, the local coverage is 90%). However,

4.6.4 It remains to rigorously characterize MTAD with imperfect coverage for tests by healthy nodes.

66. For diagnosis, we direct the edges of these chordal graphs, thus obtaining the digraphs of [Preparata et al 1967].

5. *Starchart* For a *ProtoStar* Multicomputer

How do we achieve *starship multicomputers* that satisfy the priorities of Table 3? To begin, we need to refine our *Starchart* in a fashion that pinpoints, predicts, and promotes proper technologies, models, tools, and processes. Let us elaborate the genesis of this *Starchart*. In modest detail, Section 3 identifies appropriate technologies for survivability (3.6); self-configuration (3.7, 3.8); and CAD (3.14). Section 3 also recommends directions for process (3.13, 3.15, 3.16, and 3.17). Properly speaking, our *Starchart* should also encompass key technologies for circuits. We illustrate with three examples.

Perhaps the best news for avionics is that it takes between 18 and 24 months to transform a commercially available integrated circuit (IC) into a radiation-hardened version ([Lockheed Martin 1997], [Lockheed Martin 1997 COTS], [Marshall 1999], [Marshall and Meyer 1999]). In consequence, starship avionics should be able reap most of the speed and density, and some of the economic benefits, predicted by Moore's Law.[67] Since commercial ICs are tracked and forecasted by the Semiconductor Research Association's *Technology Roadmap* ([Geppert 1999], [SRC 1998]), standardized radiation-hardening techniques reduce (but do not totally eliminate, *cf*. 3.13) the labor required to compose and maintain our *Starchart*.

In the case of starships, will need to look beyond the decade-long timeframe that is the focus of the *Roadmap*. In addition to appropriate technologies, moreover, it is incumbent upon us to identify technologies that may be *inappropriate* for use in starships. For example, the emerging field of molecular computing offers promising prospects for economical yield and reliability ([Collier et al 1999], [Tech. Rev. 1999], *cf*. 3.15). Within the next forty years, however, it does not appear that chemical computers will be ready to execute programs at a level consistent with 3.1 through 3.5.[68]

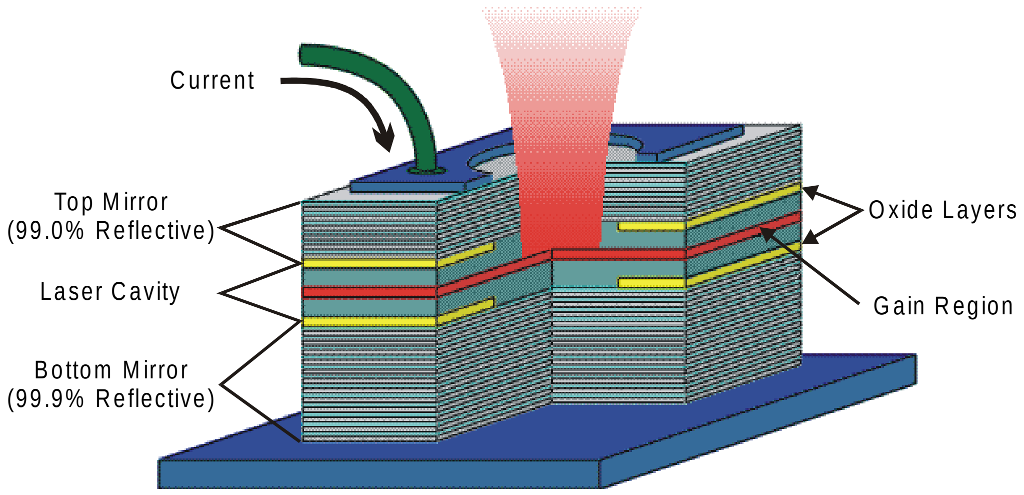


Figure 33: VCSEL: Vertical Cavity Semiconductor Emitting Laser.

By directly involving commercial manufacturers in the standardization of a *family* of computational avionics, we can ameliorate the premium we pay for space-qualified components. With respect to 3.7a and 3.7b, for example, free-space optical interconnect appears be a very viable solution in the ten to fifteen year timeframe. Refer to Figure 33. The advent of the VCSEL (Vertical Cavity Semiconductor Emitting Laser) has given rise to demonstration photonics backplanes [Guilfoyle et al 1998], [Ishikawa and McArdle

67. This time we really mean Moore's Law,[12] though some question whether it will apply in twenty or even ten years [Hamilton 1999], [Economist 1997], [Schaller 1997].
68. Admittedly, it would be *most* satisfying to live to see our prediction proved wrong.

1998], [Szymanski and Supmaonchai 1996]. As suggested in Section 3, VCSEL's also go a long way toward satisfying requirements 3.9 through 3.12. Recalling Figure 20, we can use VCSEL technology to implement alternative architectures synthesized by STAArchitecture. Figure 1 depicts a design that we hope to pursue for our Phase II effort: a prototype starship multicomputer we call *ProtoStar*.

Our Phase II *Starchart* will provide more than just a list of technologies, appropriate and inappropriate, for use in the *ProtoStar* family of multicomputers. Perhaps more importantly, our *Starchart* will gestate high-level requirements imposed by scientific, communication, and navigational aspects of interstellar missions (*cf.* 3.5). Linking high-level avionics with low-level software, our *Starchart* will, for example, bracket behavioral characteristics of workload (*cf.* p. 8) and starship instrumentation (*cf.* p. 9). For the latter, we plan allocate a portion of Phase II resources to colleagues and students at Embry-Riddle Aeronautical University[69] and at the University of Colorado. As to behavioral characteristics of workload, we are posed to benefit synergistically from *RoboComp: Roving Autonomous Astronomer on a Computer* [LaForge 2000 RoboComp]. In collaboration with SoHaR and JPL, *RoboComp* is a distinct research proposal that focuses on software for high-level autonomy (*cf.* 3.1, 3.5a).

Our Phase I effort establishes STAArchitecture version 0.5 as a vehicle that facilitates the application of architectural knowledge. For Phase II, we view STAArchitecture as an integral tool in a suite of CAD software that, in conjunction with our *Starchart*, enables the *ProtoStar* family of starship multicomputers. In addition to items 4.6.2 through 4.6.4, we hope to have the opportunity to enhance STAArchitecture in several directions; *e.g.*, automated batch runs of statistically significant simulations, with faults in switches and channels. Recalling the discussion on page 14, we also plan to incorporate a non-linear programming model that balances process-level radiation hardening, shielding, and architectural level fault tolerance.

With particular emphasis on interstellar missions, we are pleased to present positive prospects for self-healing architectures and algorithms. Presuming commensurate progress in the allied domains of Figure 3, we project and propose a Phase II effort that advances serious development of a *Starchart*, of our STAArchitecture software, and of the *ProtoStar* family of starship multicomputers exemplified in Figure 1.

A. References

[Agarwal 1991] A. Agarwal. Limits on interconnection network performance. *IEEE Transactions on Parallel and Distributed Systems*, **2** (4), October, 1991. pp. 398–412.

[Alkalai and Geer 1996] 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.

[Alkalai and Tai 1998] L. Alkalai and A. T. Tai. Long-life deep-space applications. *Computer*. August, 1998, pp. 37-38.

[Anderson 1998] D. Anderson. *Firewire System Architecture*. Reading, MA: Addison Wesley, 1998.

[Anderson 1999] J. R. Anderson. Personal communication from. Email message with attached archive, ISP.hqx, embedding file ISP.pict. 9-Dec-1999.

[Armstrong and Gray 1981] J. R. Armstrong and F. G. Gray. Fault diagnosis in a boolean *n* cube of microprocessors. *IEEE Transactions on Computers*. **C-30** (8), August, 1981. pp. 587-590.

69. Formerly a full time faculty member, principal investigator is an adjunct assistant professor at Embry-Riddle Aeronautical University.

[Artin 1975] M. Artin. *Linear Algebra and Group Theory*. Massachusetts Institute of Technology: notes for course 18.701, 1975.

[Aviation Week 1997] *Ariane 5* report details software design errors. *Aviation Week and Space Technology*. 9-Sep-1997. pp. 79-81.

[Avizienis 1967] A. Avizienis. Design of fault-tolerant computers. *1967 Fall Joint Computer Conference, AFIPS Conference Proceedings*, **31**. Washington, D.C.: Thompson-Brooks, 1967. pp. 733-743.

[Avizienis 1997] A. Avizienis. Toward systematic design of fault-tolerant systems. *Computer*. April, 1997. pp. 51-58.

[Avizienis 1999] A. Avizienis. The hundred year spacecraft. *Proceedings, First NASA/DOD Workshop on Evolvable Hardware*. Pasadena, CA: Jet Propulsion Laboratory, 19, 20, 21-July-1999. pp. 233-239.

[Banerjee et al 1986] P. Banerjee, S-Y Kuo, and W. K. Fuchs. Reconfigurable cube-connected cycles architectures. *Proceedings, 16th International Symposium on Fault Tolerant Computing*. July, 1986, pp. 286-291.

[Barry 1998] R. C. Barry. *X2000 Fault Protection Approach, Plan*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 26-Feb-1998.

[Bendetto 1998] J. M. Bendetto. Economy-class ion-defying ICs in orbit. *IEEE Spectrum*. March, 1998. pp. 36-41.

[Bernard 1999] D. E. Bernard. *RAX Update and Schedule*. Email message. 24-May-1999.

[Bianchini and Buskens 1992] R. P. Bianchini, Jr. and R. W. Buskens. Implementation of on-line distributed systems-level diagnosis theory. *IEEE Transactions on Computers*, **41** (5), May, 1992.

[Biggs 1993] N. Biggs. *Algebraic Graph Theory*. New York: Cambridge University Press, 1993. 2nd ed.

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

[Bollabás, 1978] B. Bollabás. *Extremal Graph Theory*. London: Academic Press, 1978.

[Braham 1999] R. Braham. Space: the wheel turns. In Technology 1999 analysis and forecast: aerospace and military, *IEEE Spectrum*. January, 1999. pp. 73-78.

[Bryant et al 1989] W. H. Bryant, R. K. Iyer, J. B. Dugan, W. L. Heimerdinger, J. Lala, J. Peterson, B. Randell. Computer-aided design of dependable mission critical systems. Report of a panel discussion, in Digest of Papers: 19th Symposium on Fault-Tolerant Computing Systems. Los Alamitos, CA: IEEE Computer Society Press, 1989. pp. 416-420.

[Broad 1999] W. J. Broad. Experts warn of Mars on the cheap. *New York Times*. 8-Dec-1999. p. A15.

[Butler et al 1999] R. P. Butler, G. W. Marcy, D. A. Fischer, T. W. Brown, A. R. Contos, S. G. Korzennik, P. Nisenson, and R. W. Noyes. Evidence for multiple companions to Upsilon Andromedae. *Astrophysical Journal*. Submitted, 1999. Online at http://cfa-www.harvard.edu/afoe/ and at http://www.physics.sfsu.edu/~garcy/planetsearch/upsand/upsand.html.

[Carson 1996] J. Carson. The emergence of stacked 3D silicon and its impact on microelectronics system integration. *Proceedings, IEEE International Conference on Innovative Systems in Silicon*. S. Tewksbury and G. Chapman, eds. IEEE Press, 1996. pp. 1-8.

[Cash 1999] W. Cash, personal communication with. NASA Institute for Advanced Concepts Fellows Meeting. 9-Nov-1999.

[Cash 1999 X-File] W. Cash. *X-Ray Interferometry: The Future of X-Ray Astronomy*. Viewgraph presentation. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8, 9-Nov-1999. Online at http://peaches.niac.usra.edu/library/nov99_mtng/cash_nov99.pdf.

[Cassanova 1999] R. Cassanova. Opening Remarks. Viewgraph presentation. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8, 9-Nov-1999. Online at http://peaches.niac.usra.edu/library/nov99_mtng/cassanova_nov99.pdf.

[Chartrand and Lesniak 1986] G. Chartrand and L. Lesniak. *Graphs and Digraphs*. Belmont, CA: Wadsworth, Inc, 1986. 2nd ed.

[Chau 1998] S. Chau. *X2000 Core Avionics Peer Review: Characteristics of a Reliable Bus*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 17-Apr-1998. Online at http://knowledge.jpl.nasa.gov/adssdlib/.

[Chau 1998 PDR] S. Chau. *Backup: Upstream Connection Failure.* Viewgraph presentation, X2000 Preliminary Design Review. Pasadena, CA: Jet Propulsion Laboratory, 18, 19, 20-Aug-1998.

[Chau 1999] S. Chau. Personal communication. 10-Dec-1999.

[Chau and Holmberg 1998] S. Chau and E. Holmberg. *X2000 Core Avionics Peer Review: CDH Slices Description*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 17-Apr-1998. Online at http://knowledge.jpl.nasa.gov/adssdlib/.

[Chen and Upadhyaya 1993] Y-Y Chen and S. J. Upadhyaya. Reliability, reconfiguration, and spare allocated issues in binary tree architectures based on multiple-level redundancy. *IEEE Transactions on Computers*, **42** (6), June, 1993. pp. 713–723.

[Clark 1998] D. Clark. Defect-tolerant computer. *Computer*. August, 1998. p. 27.

[Cole 1999] J. Cole. Personal communication. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8-Nov-1999.

[Collier et al 1999] C. P. Collier, E. W. Wong, M. Belohradsky, F. M. Raymo, J. F. Stoddart, P. J. Kuekes, R. S. Williams, J. R. Heath. Electronically configurable molecular-based logic gates. *Science*, **285**. 16-Jul-1999. pp. 391–394.

[Comtet 1974] L. Comtet. *Advanced Combinatorics*. Dordrecht, Holland: R. Reidel Publishing, 1974. 2nd ed.

[Culbertson et al 1996] B. Culbertson, R. Amerson, R. Carter, P Kuekes, and G. Snider. The Teramac custom computer: extending the limits with defect tolerance. *Proceedings, 1996 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*.

[Culbertson et al 1997] B. Culbertson, R. Amerson, R. Carter, P Kuekes, and G. Snider. Defect tolerance on the Teramac custom computer. *Proceedings, 1997 IEEE Symposium on FPGA's for Custom Computing Machines*. pp. 116-123.

[Das et al 1993] A. Das, K. Thulasiraman, V. K. Agarwal, K. B. Lakshmanan. Multiprocessor fault diagnosis under local constraints. *IEEE Transactions on Computers*, **42** (8), August 1993. pp. 984-988.

[Dornheim 1998] M. A. Dornheim. *Deep Space 1* launch slips three months. *Aviation Week and Space Technology*. 27-Apr-1998. p. 39.

[Dumullin 1999] J. Dumullin. *Apollo 17 (33)*. Kennedy Space Center Historical Archives. 23-Jun-1999. Online at http://www.ksc.nasa.gov/history/apollo/apollo-17/apollo-17.html.

[Dutt and Hayes 1990] S. Dutt and J. P. Hayes. On designing and reconfiguring k-fault-tolerant tree architectures. *IEEE Transactions on Computers*, **39** (4), April, 1990. pp. 490–503.

[Dvorak 1998] D. L. Dvorak. Revisions for MDS prototype. Pasadena, CA: Jet Propulsion Laboratory internal document, 7-Jul-1998.

[Dvorak 1999] D. L. Dvorak, personal communication with. 30-Nov-1999.

[Dvorak and Doyle 1998] D. L. Dvorak and R. J. Doyle. *Robo-Scientist*. Pasadena, CA: Memorandum to Jet Propulsion Laboratory Chief Scientist Moustafa Chahine, 11-Jun-1998.

[Economist 1997] Not Moore's Law. *The Economist*. 12-Jul-1997. p. 72.

[Eldred 1997] D. B. Eldred. *DS1 Flight Software Integration: Causes for Schedule Slip*. Viewgraph presentation, Pasadena, CA: Jet Propulsion Laboratory, 26-Feb-1997, revised 9-Aug-1997.

[Elfun 1977] Schenectady Elfun Society. *The General Electric Story, A Photo History, Volume 2: The Steinmetz Era, 1892–1923*. Schenectady, NY: Elfun Hall of History Publications, 1977.

[Feather 1999] M. Feather. Personal communication from. Email message. 14-Feb-1999.

[Foust 1999] J. Foust. E.T. – don't call home: a spacecraft thinks for itself. *Technology Review*. September / October, 1999. p 36.

[Gavit 1999] S. A. Gavit. *Interstellar Probe*. Viewgraph presentation 8_991ISPSummary_PublicPackage.ppt, Pasadena, CA: Jet Propulsion Laboratory, 26-Aug-1999.

[Geppert 1998] L. Geppert. The age of copper. In Technology 1998 analysis and forecast: solid state, *IEEE Spectrum*. January, 1998. pp. 23-28.

[Geppert 1999] L. Geppert. The 100-million transistor IC. *IEEE Spectrum*. July, 1999, pp. 23-24.

[Glass 1997] R. Glass. Telling good numbers from bad ones. *IEEE Software*. July/Aug-1997. pp. 15-19.

[Gluck 1999] P. R. Glück. *Remote Agent Operational (2 Day Test)*. Email message. 17-May-1999.

[Goldin et al 1998] D. S. Goldin, S. L. Venneri, and A. K. Voor. Beyond incremental change. *Computer*. October, 1998. pp. 31-40.

[Goldsmith 1999] D. Goldsmith. *Voyage to the Milky Way: The Future of Space Exploration*. New York: TV Books, 1999.

[Guiar 1998] C. N. Guiar. *X2000 High Level Driving Technical Requirements/Constraints*. Viewgraph presentation, Core Avionics Peer Review. Pasadena, CA: Jet Propulsion Laboratory, 11-Jun-1998.

[Guilfoyle et al 1998] P. S. Guilfoyle, J. M. Hessenbruch, and R. V. Stone. Free-space interconnects for high-performance optoelectric switching. *Computer*. February, 1998. pp. 69-75.

[Halvorson 1999] T. Halvorson. Faulty wire harness brought down Titan 4a rocket. *Florida Today*, Space Online: http://www.flatoday.com/space/explore/stories/1999/011699f.htm. 16-Jan-1999.

[Haverkort and Niemegeers 1996] B. R. Haverkort and I. G. Niemegeers. Performability modelling tools and techniques. *Performance Evaluation*, **25**. 1996. pp. 17-40.

[Hecht and Fiorentino 1988] M. Hecht and E. Fiorentino. Causes and effects of spacecraft failures. *Quality and Reliability Engineering International*, **4** (11), August, 1988. pp. 12-19.

[Hamilton 1999] S. Hamilton. Taking Moore's Law into the next century. *Computer*. January, 1999. pp. 43-48.

[Harris] *Harris Semiconductor Quality and Reliability Manual*. Intersil Corporation (formerly Harris Semiconductor) reliability white paper. Online at http://rel.semi.harris.com/docs/rel/.

[Hayes 1976] J. P. Hayes. A graph model for fault tolerant computing systems. *IEEE Transactions on*

Computers, **C-25** (9), September, 1976. pp. 875-884.

[Hopcroft and Ullman 1979] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Reading, MA: Addison-Wesley, 1979.

[Hotz 1999 Nov] R. L. Hotz. String of missteps doomed orbiter; JPL found at fault. *Los Angeles Times*. 11-Nov-1999. p. A1.

[Hotz 1999 Dec] R. L. Hotz. Are failed Mars probes price of cost-cutting? *Los Angeles Times* online, http://www.latimes.com/news/science/science/19991226/t000117877.html. 26-Dec-1999.

[Hunter 1997] D. J. Hunter. *X2000 Design, Implementation, and Cost Review: First Delivery Project: Electronic Packaging*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 11-through 13-Mar-1997. Online at http://knowledge.jpl.nasa.gov/adssdlib/.

[Hunter 1998] D. J. Hunter. *X2000 Peer Review: Horizontal Mounted Cube (HMC) and Integrated Avionics System (IAS), Electronic Packaging Architecture*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 16-Jul-1998.

[Irwin and Narayanan 1999] M. J. Irwin and V. Narayanan. A transition sensitive, architectural level power analysis approach. *IEEE Computer Society Technical Committee on VLSI, Technical Bulletin*. Summer 1999. pp. 6-11.

[Ishikawa and McArdle 1998] M. Ishikawa and N. McArdle. Optically interconnected parallel computing systems. *Computer*. February, 1998. pp. 61-68.

[Jezequel and Meyer 1997] J.-M. Jezequel and B. Meyer. Design by contract: the lessons of *Ariane*. *Computer*. January, 1997. pp. 129-130.

[JPL 1994 Pathfinder Software] *Mars Pathfinder AIM Flight Software*. Pasadena, CA: Jet Propulsion Laboratory internal design document, 28-Nov-1994.

[JPL DS1 Press Release] *Computer Program Assumes Spacecraft Command*. Pasadena, CA: Jet Propulsion Laboratory, Media Relations Office press release. 17-May-1999.

[JPL 1999 Voyager] *Voyager Home Page*. Pasadena, CA: Jet Propulsion Laboratory documentation, online at http://vraptor.jpl.nasa.gov/voyager/vimdesc.html. August, 1999.

[Kammash 1999] T. Kammash. *Anti-proton Driven Magnetically Insulated Inertial Fusion Propulsion System*. Viewgraph presentation. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8, 9-Nov-1999. Online at http://peaches.niac.usra.edu/library/nov99_mtng/kammash_nov99.pdf.

[Kaufmann and Freedman 1999] W. J. Kaufmann III and R. A. Freedman. *Universe*. New York: W. H. Freeman and Company, 1999. Fifth edition.

[Knowledge Adventure 1999] Knowledge Adventure. Connect the dots. Online at http://www.kidspace.com/kids/connect_dots.

[Koshgoftaar et al 1998] T. M. Koshgoftaar, E. B. Allen, R. Halstead, G. P. Trio, R. M. Flass. Using process history to predict software quality. *Computer*. April, 1998. pp. 66-72.

[Kwan and Toida 1981] C.-L. Kwan and S. Toida. Optimal fault-tolerant realizations of some classes of hierarchical tree systems. *11th Annual Symposium on Fault-Tolerant Computing*. 24-26 June, 1981. New York: IEEE Press. pp. 176-178.

[LaForge 1994] L. E. LaForge. What designers of wafer scale systems should know about local sparing. *Proceedings, Sixth Annual IEEE International Conference on Wafer Scale Integration*. Piscataway, N.J.: IEEE Press, 1994. pp 106-131.

[LaForge 1997 CVS] L. E. LaForge. *Deep Space 1*: flight software CVS logging and tagging. Pasadena, CA: Jet Propulsion Laboratory internal specification, 26-Sep-1997.

[LaForge 1997 DS1] L. E. LaForge. *Tracking and Analysis of Soft Failures and Faults for Deep Space 1*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 26-Sep-1997.

[LaForge 1999 NIAC Phase I Proposal] L. E. LaForge. *Architectures and Algorithms for Self-Healing Autonomous Spacecraft*. NASA Institute for Advanced Concepts Phase I Proposal. 31-Jan-1999.

[LaForge 1999 JPL D-16485] 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://knowledge.jpl.nasa.gov/adssdlib/ and at http://www.ec.erau.edu/cce/centers/faculty/laforgel/Current-and-Recent-Research/NASA-ASEE/.

[LaForge 1999 Trans. Computers] L. E. LaForge. Configuration of locally spared arrays in the presence of multiple fault types. *IEEE Transactions on Computers*, **48** (4). April, 1999. pp. 398-416.

[LaForge 1999 Trans. Reliability] L. E. LaForge. What designers of microelectronic systems should know about arrays spared by rows and columns. *IEEE Transactions on Reliability*. Submitted November, 1999.

[LaForge 2000 RoboComp] L. E. LaForge. *RoboComp: Roving Autonomous Astronomer on a Computer*. Notice of intent to propose. Advanced Cross-Enterprise Technology Development for NASA Missions. NRA 99-OSS-05, Topic A.5, Thinking Space Systems. 5-Jan-2000.

[LaForge 2000 Starship Avionics] L. E. LaForge. Self-healing avionics for starships. *2000 IEEE Aerospace Conference*. 18-Mar-2000.

[LaForge and Korver 2000 Graph Fault Tolerance] L. E. LaForge and K. F. Korver. Graph-theoretic fault tolerance for spacecraft bus avionics. *2000 IEEE Aerospace Conference*. 18-Mar-2000

[LaForge and Korver 2000 MTAD] L. E. LaForge and K. F. Korver. Mutual test and diagnosis: architectures and algorithms for spacecraft avionics. *2000 IEEE Aerospace Conference*. 18-Mar-2000.

[LaForge et al 1993] L. E. LaForge, K. Huang, and V. K. Agarwal. Almost sure diagnosis of almost every good element. *IEEE Transactions on Computers*, **43** (3). March, 1994. pp. 295-305.

[LaForge et al 1999] L. E. LaForge, C. Cassell, M. Hecht, K. F. Korver, D. D. Carlson. *Avionics and Software for a Starship to Alpha Centauri*. Viewgraph presentation Avionics-and-Software-for-a-Mission-to-Alpha-Centauri.ppt. Pasadena, CA: Jet Propulsion Laboratory, 25-Aug-1999.

[Lalli 1998] V. R. Lalli. Space-system reliability: a historical perspective. *IEEE Transactions on Reliability*, **47** (3). September, 1998. pp. 355-360.

[LaPointe 1999] M. R. LaPointe. *Primary Propulsion for Piloted Deep Space Exploration*. Viewgraph presentation. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8, 9-Nov-1999. Online at http://peaches.niac.usra.edu/library/nov99_mtng/lapointe_nov99.pdf.

[Larson and Wertz 1995] W. J. Larson and J. R. Wertz. *Space Mission Analysis and Design*. Boston: Kluwer Academic Publishers, 1995.

[Leighton and Leiserson 1985] T. Leighton and C. E. Leiserson. Wafer-scale integration of systolic arrays. *IEEE Transactions on Computers*, **34** (5). May, 1985. pp. 448-461.

[Lee and Cong 1997] T-C Lee and J. Cong. The new line in IC design. *IEEE Spectrum*. March, 1997. pp. 52-58.

[Lewis and Postol 1997] G. N. Lewis and T. A. Postol. Future challenges to ballistic missile defense. *IEEE Spectrum*. September, 1997. pp. 60-68.

[Lockheed Martin 1997] *The RAD6000 Flight Computer*. Technical product description. Lockheed Martin Corporation. 29-Aug-1997.

[Lockheed Martin 1997 COTS] *Radiation Hardened COST-based 32-bit Microprocessor*. Technical product description. Lockheed Martin Corporation. 1997.

[Lockheed Martin 1999] *Rad-Lite Product Discussion*. Viewgraph presentation B5346.ppt. Lockheed Martin Corporation. January, 1999.

[Lowry 1998] M. R. Lowry. Component-based reconfigurable systems. *Computer*. April, 1998. pp. 44-45.

[Marcopulos 1998] T. Marcopulos. Faster, better, cheaper space exploration. *IEEE Spectrum*. August, 1998. pp. 68-74.

[Marcus 1998] S. J. Marcus. Grand illusion: the risks of *Cassini*. *IEEE Spectrum*. March, 1998. pp. 58-65.

[Marshall 1999] J. Marshall. Personal communication. 8-Oct-1999.

[Marshall and Meyer 1999] J. Marshall and D. Meyer. Personal communication. 12-Dec-1999.

[McDermid and Talbert 1998] J. McDermid and N. Talbert. The cost of COTS. *Computer*. June, 1998. pp. 46-52.

[McNutt et al 1997] R. L. McNutt, Jr, R. E. Gold, E. C. Roelof, L. J. Zanetti, E. L. Reynolds, R. W. Farquhar, D. A. Gurnett, and W. S. Kurth. A SOLE/AD ASTRA: from the sun to the stars. *Journal of the British Interplanetary Society*, **50**. 1997. pp. 463-474.

[McNutt 1999] R. L. McNutt, Jr. *A Realistic Interstellar Explorer*. NASA Institute for Advanced Concepts Phase I final report, May, 1999. Online at www.niac.usra.edu/studies/9801/9801Final/McNuttFinal.pdf.

[Meyer 1999] B. Meyer. Every little bit counts: toward more reliable software. *Computer*. November, 1999. pp. 131-133.

[Milky Way 1999] *Voyage to the Milky Way*. Video documentary, Thomas Lucas Productions, 1999.

[Millis 1997] M. G. Millis. The challenge to create the space drive. *Journal of Propulsion and Power*. **13** (5), September / October, 1997. pp. 577 - 682. Online at http://www.lerc.nasa.gov/www/bpp/TM-107289.htm.

[Moore and Shannon 1956] E. F. Moore and C. E. Shannon. Reliable circuits using less reliable relays, part I. *Journal of the Franklin Institute*, **262**. September, 1956. pp. 191-208.

[Morrison and Schmidt 1999] D. Morrison and G. K. Schmidt, editors. *Astrobiology Roadmap*. NASA Ames Research Center document. 4-Jan-1999. Online at http://astrobiology.arc.nasa.gov/roadmap/.

[Muirhead 1996 Pathfinder Design] B. K. Muirhead. Mars Pathfinder flight system design and implementation. *Proceedings, 1996 IEEE Aerospace Conference*, **4**. Snowmass at Aspen: 1-8 Feb 1996. pp. 191-205.

[Muirhead 1996 Pathfinder Test] B. K. Muirhead. Mars Pathfinder flight system integration and test. *Proceedings, 1996 IEEE Aerospace Conference*, **4**. Snowmass at Aspen: 1-8 Feb 1996.

[Munson and Werries 1996] J. C. Munson and D. S. Werries. Measuring software evolution. *Proceedings, 1996 IEEE International Software Metrics Symposium.* IEEE Computer Society Press. pp. 41-51.

[Muraldi 1990] F. Muraldi. *A New Procedure for Weighted Random Built-in Self Test.* Ph.D. dissertation, Montreal: Department of Electrical Engineering, McGill University, March, 1990.

[Murty and Vijayan 1964] U. S. R. Murty and K. Vijayan. On accessibility in graphs. *Sakhya* Ser. A, Vol

26, 1964. pp. 299 - 302.

[Nabli and Sericola 1996] H. Nabli and B. Sericola. Performability analysis: a new algorithm. *IEEE Transactions on Computers,* **45** (4). April, 1996. pp. 491-494.

[NASA 1999] High integrity systems and algorithms. Topic 1.04, in *SBIR: Small Business Innovation Research, 1999 Program Solicitation*. National Aeronautics and Space Administration. Opening date: 24-Apr-1999.

[Neumann 1996] P. G. Neumann. Using formal methods to reduce risk. *Communications of the ACM*, **39** (7). July, 1996. p. 114.

[NIAC 1999 Agenda] NASA Institute for Advanced Concepts. *NIAC Fellows Meeting Agenda*. 8, 9-Nov-1999. Online at http://www.niac.usra.edu/library/nov99_mtng/nov99_agenda.pdf.

[NIAC 1999 Scope] NASA Institute for Advanced Concepts. *Scope of NIAC Call for Proposals*. Online at http://peaches.niac.usra.edu/library/challenges/challenges-final.pdf.

[Nikora et al 1998] A. P. Nikora, N. F. Schneidewind, and J. C. Munson. IV&V issues in achieving high reliability and safety in critical control system software, final report. Pasadena, CA: Jet Propulsion Laboratory, document number D-15740, 19-Jan-1998.

[Nordley 1998] G. Nordley. Don't rule out interstellar probes. SETI League Publications. 28-Feb-1998. Online at http://seti1.setileague.org/articles/probes.htm.

[Norvig 1999] P. Norvig. Personal communication. 2-Aug-1999.

[Oberg 1998] J. Oberg. Shuttle-Mir's lessons for the International Space Station. *IEEE Spectrum*. June, 1998. pp. 28-37.

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

[Pedder 1993] D. J. Pedder. Interconnection technologies for multichip assemblies (ITMA) – a UK information technology engineering directorate hybrid wafer scale project. *Proceedings, Fifth Annual IEEE International Conference on Wafer Scale Integration.* Piscataway, N.J.: IEEE Press, 1993. pp. 95-106.

[Pickering 1999] K. A. Pickering. *A Christopher Columbus Timeline*. 12-Oct-1999. Online at www1.minn.net/~keithp/cctl.htm.

[Preparata et al 1967] F. Preparata, G. Metze, and R. Chien. On the connection assignment problem of diagnosable systems. *IEEE Transactions on Computers*, **EC-16** (6), December, 1967. pp. 848-854.

[Preparata and Vallemin 1981] F. P. Preparata and J. Vallemin. The cube-connected cycles, a versatile network for parallel computation. *Communications of the ACM*, May 1981. pp. 30-39.

[Rasmussen and Sacks 1998] R. Rasmussen and A. Sacks. *First Delivery DICR: Mission Data System (MDS)*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 2-Mar-1998.

[Recer 1999] P. Recer. Astronomers find six new planets. *Los Angeles Times* online: http.www.latimes.com. 30-Nov-1999.

[Relex 1998] Advertisement for Relex® reliability software. Relex Software Corporation. *IEEE Reliability Society Newsletter*. April, 1998. p 7.

[Richard 1999] E. D. Richard. Saturn V rocket. In *Bridger's Space Archives*. Online at http://www.geocities.com/CapeCanaveral/3877/saturn.htm.

[Runyon 1999] S. Runyon. Testing big chips becomes an internal affair. *IEEE Spectrum*. April, 1999. pp. 49-55.

[Sampels 1997] M. Sampels. Large networks with small diameter. In *Graph-Theoretic Concepts in Computer Science, 23rd International Workshop*. Berlin: 1997. R. H. Möhring, editor. pp. 288-302.

[Savino 1997] J. L. Savino. *DS1 Lessons Learned – DS1 /Autonomy FSW.* Pasadena, CA: Jet Propulsion Laboratory internal memorandum. 14-May-1997.

[Schaller 1997] R. R. Schaller. Moore's Law: past, present, and future. *IEEE Spectrum*. June, 1997. pp. 53-59.

[Scheinerman 1987] E. R. Scheinerman. Almost sure fault tolerance in random graphs. *SIAM Journal of Computing*, **16** (6), December, 1987. pp. 1124-1134.

[Schneider 1999 E1] F. L. Schneider. Personal communication from. Email message. 15-Feb-1999.

[Schneider 1999 E2] F. L. Schneider. Personal communication from. Email message. 21-Feb-1999.

[Siewiorek and Swarz 1982] D. P. Siewiorek and R. S. Swarz. *The Theory and Practice of Reliable Design*. Bedford, MA: Digital Press, 1982.

[Slough 1999] J. R. Slough. *Propagating Magnetic Wave Accelerator for Manned Deep Space Missions*. Viewgraph presentation. Atlanta: NASA Institute for Advanced Concepts Fellows Meeting, 8, 9-Nov-1999. Online at http://peaches.niac.usra.edu/library/nov99_mtng/slough_nov99.pdf.

[Somani and Agarwal 1992] A. K. Somani and V. K. Agarwal. Distributed diagnosis algorithms for regular interconnected structures. *IEEE Transactions on Computers*, **41** (7). July, 1992. pp. 899-906.

[SRC 1998] *International Technology Roadmap for Semiconductors: 1998 Update*. Semiconductor Research Corporation. 1-Jun-1999. Online at http://www.itrs.net/ntrs/PublNTRS.nsf/lookup/98Update/$file/Update.htm.

[Stapper 1992] C. H. Stapper. *Development of IBM's 16MBit Dynamic Random Access Memory Chip.* Address to the Department of Electrical Engineering and Computer Science, University of Vermont, Burlington, VT, April, 1992.

[Stein and Little 1997] R. M. Stein and J. Little. 'Future challenges to ballistic missile defense' paints wrong picture. *IEEE Spectrum*. November, 1997. pp. 70-72.

[Steiner 1997] C. Steiner. *X2000 Design, Implementation, and Cost Review: First Delivery Project: Avionics System Engineering*. Viewgraph presentation. Pasadena, CA: Jet Propulsion Laboratory, 11-through 13-Mar-1997. Online at http://knowledge.jpl.nasa.gov/adssdlib/.

[Steiner 1998] C. Steiner. *X2000 Core Avionics Peer Review: Avionics Overview*. Viewgraph presentation. Jet Propulsion Laboratory, Pasadena, CA, 17-Apr-1998. Online at http://knowledge.jpl.nasa.gov/adssdlib/.

[Szymanski and Supmaonchai 1996] T. H. Szymanski and B. Supmaonchai. Reconfigurable computing with optical backplanes - an economic argument for optical interconnects. In *Proceedings, Third International Conference on Massively Parallel Processing Using Optical Interconnects (MPPOI '96)*. Maui, HW: 27, 28, 29-Oct-1996. pp. 321-328.

[Tang et al 1998] D. Tang, M. Hecht, J. Miller, and J. Handal. MEADEP: a dependability evaluation tool for engineers. *IEEE Transactions on Reliability*, **47** (4). December, 1998. pp. 443-450.

[Taur 1999] Y. Taur. The incredible shrinking transistor. *IEEE Spectrum*. July, 1999. pp. 25-34.

[Tech. Rev. 1999] Computing after silicon. *Technology Review*. September / October, 1999. pp. 93–96.

[Tucker 1984] Alan Tucker. *Applied Combinatorics.* New York: John Wiley and Sons, 1984.

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

[Virgras] W. J. Virgras. *Calculation of Semiconductor Failure Rates*. Intersil Corporation (formerly Harris Semiconductor) reliability white paper. Online at http://rel.semi.harris.com/docs/rel/.

[Voas et al 1997] J. Voas, G. McGraw, L. Kassab, L. Voas. A 'crystal ball' for software liability. *Computer*. June, 1997. pp. 29-36.

[Wade 1999] M. Wade. *Voyager*. In *Encyclopedia Aeronautica*. Online at http://solar.rtd.utk.edu/~mwade/project/voyager.htm. 11-Jan-1999.

[Wilford 1999] J. N. Wilford. NASA vows stiff rethinking of Mars exploration program. *New York Times*. 8-Dec-1999. p. A15.

[Woerner and Lehman 1995] D. F. Woerner and D. H. Lehman. 'Faster, better, cheaper' technologies used in the attitude and information subsystem for the *Mars Pathfinder* mission. *Proceedings, 1995 IEEE Aerospace Applications Conference*.

[Zargham 1996] M. R. Zargham. *Computer Architecture: Single and Parallel Systems*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[Zorian 1999] Y. Zorian. Testing the monster chip. *IEEE Spectrum*. July, 1999. pp. 54-60.

[Zorian et al 1999] Y. Zorian, E. J. Marinissen, and S. Dey. Testing embedded-core-based system chips. *Computer*. June, 1999. pp. 52-60.