The Roots of the Argonne Training Program on Extreme Scale Computing

Paul Messina, Ret., ANL Associate

Argonne Distinguished Fellow

August 4, 2022

Outline

- **Precursors to ATPESC**
- **Creating ATPESC**
  - The concept
  - Getting it funded
  - The curriculum
- **CSE Curriculum thoughts**
- **Random topics and reading suggestions**

The History of the Argonne Computing Research Facility - *the inspiration*

Paul Messina

Argonne Leadership Computing Facility

Argonne National Laboratory

Symposium on Thirty Years of Parallel Computing at Argonne

May 14, 2013

Parallelism was in the air 1970s to early 1980s

- **ILLIAC IV**
  - Some Argonne researchers had access to it in the late 1970s
- **Jack Schwartz and the NYU Ultracomputer project**
- **Geoffrey Fox and the Caltech Concurrent Computation Program**
- **Dave Kuck and the CEDAR project at UIUC**
- **The turning point for me: a DOE Applied Mathematical Sciences PI meeting in Germantown, MD When Jack Schwartz, NYU Courant Institute, talked about parallel computer architectures**

Let's start a facility with heterogeneous architectures

- **Jack Dongarra, Danny Sorensen, Rusty Lusk, Ross Overbeek – and others – suggested we establish the Experimental Computing Facility for MCS and other interested researchers to experiment with parallel computing**
- **Walter Massey, Argonne Lab Director, and Ken Kliewer, ALD, funded an LDRD proposal to get our own Denelcor HEP**
- **Soon after, we conceived a national facility that would host computers with a variety of architectures**
  - No clear dominant architecture
  - Develop portable approaches
- **Don Austin, then Head of DOE's AMS office, was receptive to a proposal to start a research program on parallel computing and establish such a facility, and he funded it: the ACRF was born!**

Advanced Computing Research Facility: 1984 -1992

- **The Advanced Computing Research Facility (ACRF) was established in recognition of the role that parallel computers would play in the future of scientific computing.**
- **Principal objectives:**
  - To encourage experimentation on computers with innovative designs
  - To assess the suitability of diverse machines for specific applications
  - To assist research in parallel computation
  - To encourage the incorporation of state-of-the-art computational techniques in research, development and industry.
  - To provide leadership in enhancing computing environments
  - To operate as a national user facility in parallel computing

There were lots of interesting machines …

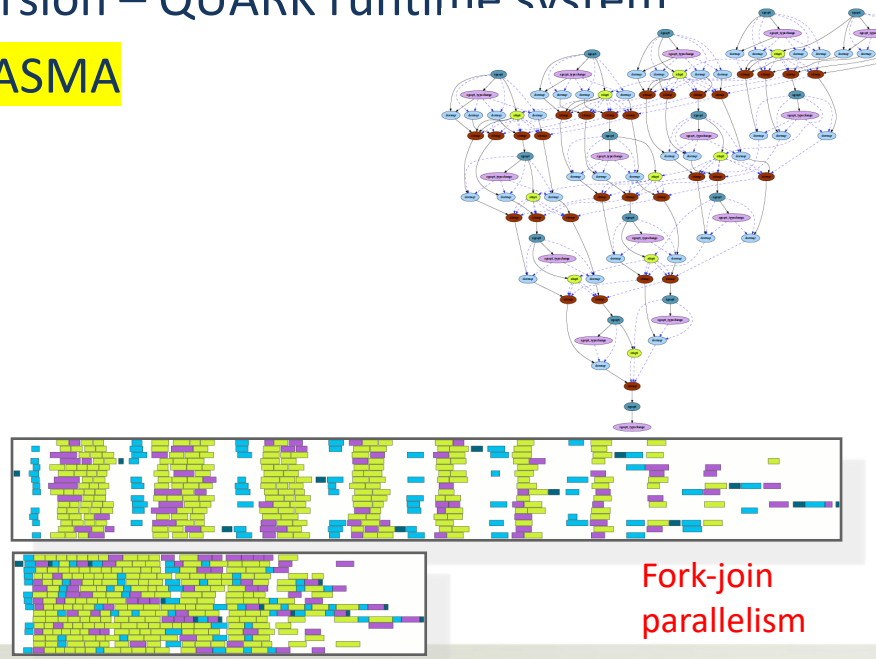
But - How to Program Them?

[MPI (1994), OpenMP (1997) did not exist yet]

- **Little Practical Experience at this point in time**
- **Some theoretical work**
  - Per Brinch Hansen's monitors
  - Tony Hoare's CSP (communicating sequential processes)
  - Gap between theory and practice persisted

Some Early Portable Programming Models (Not Dead Yet!)

- **The Jack Dongarra – Danny Sorensen approach: SCHEDULE**
  - User defines dependency graph. Each unit of computation is a Fortran subroutine.
  - User required to specify each unit of computation, number of tasks below in the graph and the identity of the ones above.
  - Given the dependency information, SCHEDULE takes over and schedules the work.
  - Designed to be portable and easy to bring up on a new system.
- Current Version – QUARK runtime system
- Used in PLASMA



Fork-join parallelism

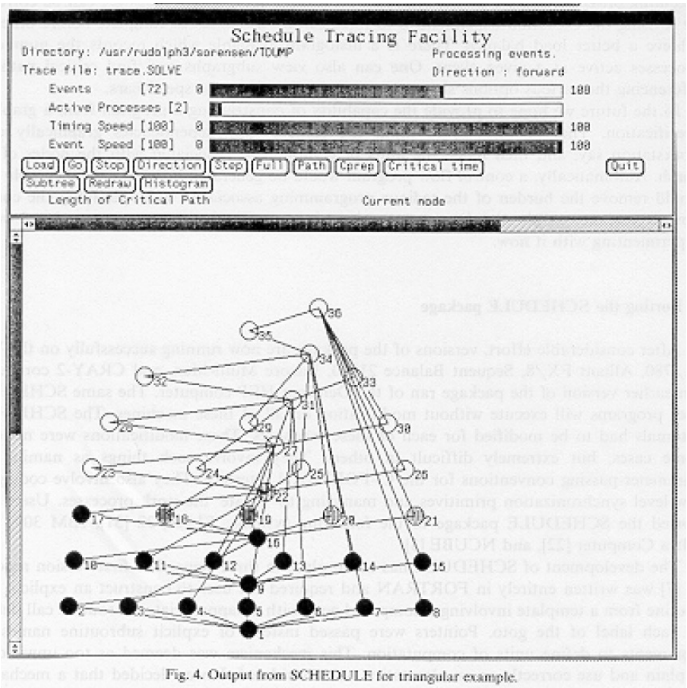


Fig. 4. Output from SCHEDULE for triangular example.

Some Early Programming Models (cont.) (Not Dead Yet!)

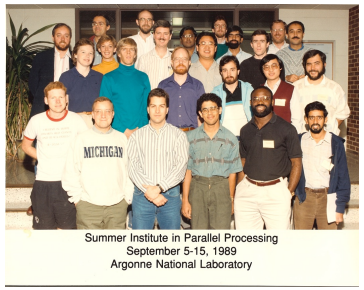
- The Ross Overbeek – Rusty Lusk approach: **monmacs**
  - Use vendor-specific concepts to implement locks
  - Use locks to implement monitor-building primitives (enter, exit, delay, continue)
  - Use these primitives to implement a library of useful monitors
  - Implement with (m4) macros for performance
  - Most useful was the askfor monitor: managing a shared work pool without master-slave logic
- Current version: ADLB (Asynchronous Dynamic Load Balancing) Library
  - scales to a million processes
  - used in nuclear physics application
  - put/get for shared work pool of typed tasks
  - can be used to implement multiple algorithms

Outreach

- Sensing a growing interest in these parallel computing approaches, we decided to offer courses on how to program parallel computers
- Two-week summer institutes
- 3-day courses every six weeks

The ACRF Summer Institutes (Not Dead Yet!)


NSF 1987
Summer Institute of Parallel Computing



Summer Institute in Parallel Processing
September 5-15, 1989
Argonne National Laboratory

- Sponsored by NSF
- Held each September, 1987-1989
- Mornings: long lectures by distinguished speakers
- Afternoons: hands-on experience with Argonne software on ACRF machines
- Some speakers: Gordon Bell, Bill Buzbee, Josh Fisher, Dave Kuck, Neil Lincoln, Chuck Seitz, Larry Smarr, Burton Smith, Guy Steele, Don Austin, Mani Chandy, Arvind, Tom DeFanti, David Gelernter, John Gurd, Ken Kennedy, Alex Nicholau
- Admission by application and review, about 20 students each summer
- Next Summer Institute: July 29 – Aug 9, 2013 [1st ATPESC]

The High-Volume Approach: Regular Courses

- Every six weeks, Argonne held a 3-day course in parallel computing free and open to anyone.
- Gave people experience on ACRF machines
- Conveyed Argonne portable programming models and ideas
- Introduced a whole generation to parallel computing

Impact of the ACRF courses: Career of two participants

- **Barb Helland and Bill Harrod both told me that attending Argonne's parallel computing courses in mid 1980s changed their careers**
- **Barbara Helland**
  - Associate Director of the DOE Office of Science for Advanced Computing Research
  - Program Manager for Exascale Computing Project
  - Program Manager for ALCF and OLCF
  - Krell Institute
  - DOE Ames Lab computational scientist
- **Wiliam Harrod**
  - IARPA Program Manager 2018 – present
  - Co-Director, Joint Program Office for Strategic Computing, DOE, 2017-2018
  - Research Division Director, for Advanced Computing Research, DOE
  - Program Manager, DARPA 2005 -2010
  - SGI 1996-2005
  - Cray Research 1991-1996
- **CAVEAT: Historical performance is not a guarantee of future results**

An Additional Model: Grid Summer Schools

- **In the early 2000s for grid computing, a precursor of cloud computing, was being adopted but there were no formal courses on how to set up a grid and use its software infrastructure**
- **Several colleagues and I organized and held international two-week summer schools that covered the relevant topics and had substantial hands-on exercises.**
  - A CERN summer school was a partial model
- **The grid schools were funded by the NSF and the European Union, based on an unsolicited proposal**

Grid Computing School 2003

held in Sorrento Peninsula, tough duty

Curriculum for
International Summer School on Grid Computing

As of 22July03

Sunday, July 13, 2003	
20:30	**Opening Reception and Dinner at Hotel**
Monday, July 14, 2003	
08:30 – 09:00	**Welcoming Remarks** Prof. Almerico Murli, University of Naples Federico II Dr. Mario Montanino, CNR Institute of Composite and Biomedical Materials Dr. Fabrizio Gagliardi, CERN Dr. Paul Messina, ANL & CERN, Chair of Organizing Committee
09:00 – 13:00	**Globus Toolkit 2, Condor G** Presented by: Carl Kesselman, Miron Livny
14:30	**Lab Exercises**
21:00	**Pompeii lecture** Presented by: TBD
Tuesday, July 15, 2003	
08:30 – 13:00	**European DataGrid Tutorial** Presented by: Heinz Stockinger, Erwin Laure, Laurence Field
14:30	**Lab Exercises**
21:00	**Lecture: CMS Production Use of the EDG Testbed** Presented by: Alessandra Fanfani
Wednesday, July 16, 2003	

The Motivation for ATPESC

- As Director of Science for the ALCF, I noticed in 2012 that many users of high-end computer systems, such as ALCF supercomputers, lacked the expertise required to use them effectively and some were even unaware of the existence of debugging tools.
- Having been deeply involved in activities related to developing and using exascale systems, I knew that computer architectures would become more complex, as would the applications that would require exascale computing power.
- I was inspired to start a summer school by my recollection of the ACRF short courses on parallel programming and the grid schools

Taking action

- **I mentioned to some Argonne colleagues that I wanted to create a summer school for high-end computational science, and they agreed that it would fill an important gap.**
- **I then pitched the idea to two Program Managers in DOE's Office of Advanced Scientific Computing Research (ASCR)**
- **They thought it was an interesting idea and suggested I submit a formal proposal**
- **I wrote a proposal for funding the first three years of the ATPESC**
  - Requested $1.2M

The Outcome

- The proposal was sent for peer review and the reviews were positive enough to convince the program managers to fund it
- DOE funding covers the cost of your travel, lodging, the Q Center facilities
- Argonne divisions contribute staff and lecturer time … a lot of time!

Key Features

- **In person (except for COVID years)**
  - No MOOCS!
- **Moderate number of participants to foster interaction**
- **Hands-on sessions on leading-edge computers**
- **Lecturers who are leaders in their fields**
- **Broad curriculum**
  - Currently you might not need some of the tools or methods that are presented but in the future when you might, you will have a starting point of what to consider, in topics such as software engineering, visualization software, debugging and performance measurement tools

How ATPESC took shape and evolved

- **I would have liked to make it a one month school (or longer)**
- **Could not use the words "school" or "exascale" in the title**
- **Initial curriculum was defined by a committee**
  - Pete Beckman, Richard Coffey, Rusty Lusk, Paul Messina, Michael Papka, Katherine Riley, and Rajeev Thakur
- **High-level curriculum has remained but much evolution in the specific topics covered … and computers used**
  - E.g., AI
- **Evolution of attendees in 10 years**
  - More computer scientists in 2022, initially we selected **computational** scientists almost exclusively

As in the 1980s, A Plethora of New Computer architectures

- **2017 architectures track covered OLCF systems Summit and Frontier, ARM (LANL and Sandia), quantum, FPGAs and machine learning. Hands-on systems: KNL (Theta), BG/Q (Mira), Cray XK7 (Titan), KNL (Cori).**
- **2020 program included**
  - talks by M. Sato, RIKEN, on Fugaku, on Cerebras by Rob Schreiber;
  - track 8 was on machine learning; systems used were Frontier, Aurora (?), Perlmutter, El Capitan [need to check on whether all were used hands-on, what “Aurora” was.]
- **2021 program architecture track featured many systems:**
  - Cerebras,
  - Frontier,
  - Groq,
  - HPE,
  - Sambanova,
  - Havana (an Intel company),
  - Perlmutter,
  - Aurora (Intrepid to Aurora),
  - Quantum.

First ATPESC July 29 – August 9, 2013

Anecdotes from my time in ATPESC

- I particularly enjoyed seeing graduate students and postdocs asking intelligent questions of lecturers who were renowned in their fields, and witnessing students deeply engaged in the hands-on exercises – an opportunity that was hard to replicate.
- Also, seeing participants working side by side with a lecturer to use new tools to debug or analyze the program they use for their research.


Aditya Konduri
pepsi

Anecdotes from my time in ATPESC

- I remember having a participant from a DOE national laboratory with 20 years of experience in computing tell me at the end of the course that he had learned many useful things, was glad he attended, and he would encourage others from his lab to apply for future editions
- Learning with pleasure that two graduate students who participated in the first ATPESC had papers accepted at the following year's Supercomputing Conference, the premier conference in this field and one with a low acceptance rate for papers. They credited having attended ATPESC for their success.

Thoughts on curricula for Computational Science and Engineering (CSE)

- In early 2015 I had planned to phase out of my role as Director of Science for the ALCF at the end of the summer and focus on establishing relationships with universities and working with them on curricula for computational science and engineering degrees
- In July 2015 DOE HQ asked me to launch and lead the US Exascale Computing Project and I accepted the invitation, so unfortunately I was not able to pursue the university curricula activities

Fortunately, others worked - and work - on CSE curricula

- **E.g., the article based on the report from a workshop sponsored by the Society for Industrial and Applied Mathematics (SIAM) and the European Exascale Software Initiative (EESI-2), August 4--6, 2014, Breckenridge, CO.**

Report from a workshop sponsored by the Society for Industrial and Applied Mathematics (SIAM) and the European Exascale Software Initiative (EESI-2), August 4--6, 2014, Breckenridge, CO.

SIAM REVIEW
Vol. 60, No. 3, pp. 707–754

© 2018 Society for Industrial and Applied Mathematics

Research and Education in Computational Science and Engineering*

Officers of the SIAM Activity Group on Computational Science and Engineering (SIAG/CSE), 2013–2014:
Ulrich Rüde†
Karen Willcox‡
Lois Curfman McInnes§
Hans De Sterck¶

Learning outcomes desired of a student graduating from a CSE Ph.D. program (1)

CSE Ph.D. with Broadly Applicable CSE Focus	CSE Ph.D. with Domain-Driven Focus in Field X
Combine mathematical modeling, physical principles, and data to derive, analyze, and assess *models across a range of systems (e.g., statistical mechanics, continuum mechanics, quantum mechanics, molecular biology).*	Combine mathematical modeling, physical principles and data to derive, analyze, and assess a *range of models within field X.*
Demonstrate graduate-level depth in devising, analyzing, and evaluating new methods and algorithms for computational solution of mathematical models (including parallel, discrete, numerical, statistical approaches, and mathematical analysis).	
Demonstrate *mastery* in code development to exploit parallel and distributed computing architectures and other emerging modes of computation in algorithm implementation.	Demonstrate *proficiency* in code development to exploit parallel and distributed computing architectures and other emerging modes of computation in algorithm implementation.
Be aware of available tools and techniques from software engineering, their strengths, and their weaknesses; select and apply techniques and tools from software engineering to build robust, reliable, and maintainable software.	

Learning outcomes desired of a student graduating from a CSE Ph.D. program (2)

Develop, select, and use tools and methods to represent and visualize computational results.	
Critically analyze and evaluate results using mathematical and data analysis, physical reasoning, and algorithm analysis, and understand the implications on models, algorithms, and implementations.	
Identify the sources of errors in a CSE simulation (such as modeling errors, code bugs, premature termination of solvers, discretization errors, roundoff errors, numerical instabilities), and understand how to diagnose them and work to reduce or eliminate them.	
Appreciate and explain the context of decision-making as the end use of many CSE simulations, and as appropriate be able to formulate, analyze, and solve CSE problems in control, design, optimization, or inverse problems.	Appreciate and explain the context of decision-making as the end use of many CSE simulations, and as appropriate be able to formulate, analyze and solve CSE problems in control, design, optimization or inverse problems *as relevant to field X*.
Understand data as a core asset in computational research, and demonstrate appropriate proficiencies in processing, managing, mining, and analyzing data throughout the CSE/simulation loop.	

Learning outcomes desired of a student graduating from a CSE Ph.D. program (3)

Demonstrate the ability to develop, use, and analyze sophisticated computational algorithms in data science and engineering, and understand data science and engineering as a novel field of application of CSE.	
Demonstrate graduate-level *proficiency in one domain in science or engineering.*	Demonstrate graduate-level *depth in domain knowledge in field X.*
Communicate across disciplines and collaborate in a team.	

I recommend reading that report

https://epubs.siam.org/doi/abs/10.1137/16M1096840

Some random topics

Unsolicited Proposals are funded (sometimes)

- **The ACRF**
- **The Grid Schools**
- **The ATPESC**

And numerous others, e.g., Charlie Catlett's Smart Cities project that Pete Beckman mentioned in his talk

- **I encourage you to consider submitting unsolicited proposal**
  - If possible, pitch your ideas informally first to avoid wasting time on a full formal proposal

When was the first Nobel Prize awarded that relied crucially on Computational Science?

John Kendrew's 1962 Nobel Prize

- **The EDSAC (Electronic Delay Storage Automatic Calculator), an early British computer inspired by John von Neumann's seminal "First Draft of a Report on the EDVAC," was working in Cambridge before EDVAC (Electronic Discrete Variable Automatic Computer).**
- **EDSAC was used by John Kendrew in for research that was awarded the 1962 Nobel Prize in Chemistry (jointly with Max Perutz). In collaboration with John Bennett, Kendrew produced the first-ever program for computing a Fourier Summation for X-ray structure analysis.**
  - Kendrew determined the structure of myoglobin, a 2,600 atom protein that carries oxygen to muscles, by using EDSAC to determine its three-dimensional structure by examining 110 crystals and measuring the intensities of around 250,000 X-ray reflections.
  - Data intensive for the 1950s!

Vannevar Bush's Essay "As We may Think" published in *The Atlantic* in 1945

- **Now that WWII has ended, he urges that scientists should turn to the massive task of making more accessible our bewildering store of knowledge. For years inventions have extended man's physical powers rather than the powers of his mind. …. Now, says Dr. Bush, instruments are at hand which, if properly developed, will give access to and command over the inherited knowledge of the ages.**
- **In this essay, Bush predicted many kinds of technology invented long after its publication, including hypertext, personal computers, the Internet, the World Wide Web, speech recognition, and online encyclopedias such as Wikipedia. The vision contained in that article is awe inspiring.**

- **Vannevar Bush is best known for his July 1945 report to the the US President "Science: the Endless Frontier"**

President F.D. Roosevelt's November 17, 1944 charge to V. Bush that resulted in "Science: the Endless Frontier"

- (1) What can be done, consistent with military security, and with the prior approval of the military authorities, to make known to the world as soon as possible the contributions which have been made during our war effort to scientific knowledge?
- (2) With particular reference to the war of science against disease, what can be done now to organize a program for continuing in the future the work which has been done in medicine and related sciences?
- (3) What can the Government do now and in the future to aid research activities by public and private organizations?
- (4) Can an effective program be proposed for discovering and developing scientific talent in American youth so that the continuing future of scientific research in this country may be assured on a level comparable to what has been done during the war?

You might consider reading those two documents by V. Bush

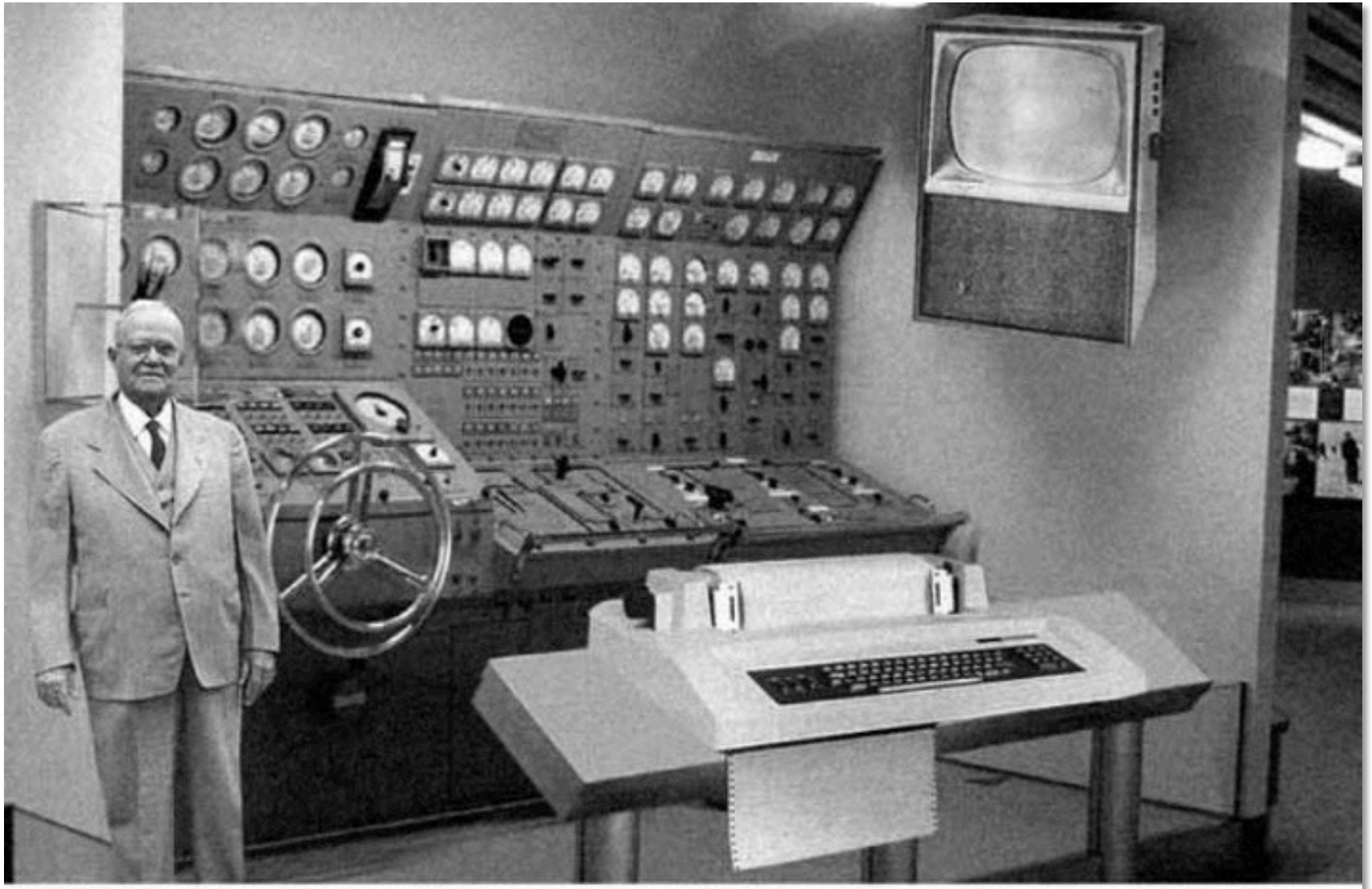
Not all predictions of future technologies are accurate

Notional Architecture Trends

[from my 2013 Intro to ATPESC talk]

Systems	2012	2017 +1/-0	2022 +1/-0
System peak	**20 Peta**	**100-300 Peta**	**1 Exa**
Power	**10 MW**	**~15 MW**	**~20 MW**
Node concurrency	12	O(100)	O(1k) or 10k
Total Node Interconnect BW	3.5 GB/s	100-200 GB/s 10:1 vs memory bandwidth 2:1 alternative	200-400GB/s (1:4 or 1:8 from memory BW)
System size (nodes)	18,700	50,000 or 500,000	O(100,000) or O(1M)
Total concurrency	225,000	O(100,000,000) *O(10)-O(50) to hide latency	O(billion) * O(10) to O(100) for latency hiding

2004 Home Computer Imagined in mid 1950s



Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 2004. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.

Feedback

- I know you are asked for feedback on ATPESC but I would appreciate any comments and suggestions you have for improvements, based on your experience so far.

Thanks for participating in the 2022 ATPESC!