Javaslat

tiszteletbeli doktori cím adományozására

Név: **Bélády László**
Született Budapesten, 1928 április 29
Tudományos képesítés ill. minősítés: gépészmérnök, a MTA külső tagja (VI. Osztály, 1990)
Állampolgársága: magyar, USA
Lakcím: 1805 Canonero Drive, Austin TX, 78746, USA ill.
H-1052 Petőfi Sándor-u. 11, Budapest
Munkahelye: Univ. of Austin ill. EUTECUS Inc., President
Rövid életrajz:

1950: repülőgép-mérnöki diploma a BME Gépészmérnöki Karán

Föbb munkahelyek:

1950-53: Magyar Légierő
1953-56: Középgépipari Kutató Intézet
1956-59: Ford Motor Company, Köln (*Design Engineer*)
1959-61: Dassault Aeronautics, Paris (*Design Engineer*)
1961-81: IBM Watson Research Center, Yorktown Heights (*Research Manager*)
1981-83: IBM Corporate Headquarters, Armonk (*Corporate Director of Software Technology)*
1983-84: IBM Japan Research Center, Tokyo *(General Manager)*
1984-91: Microelectronics & Computer Technology Corp (MCC), Austin (*Vice President)*
1991-98: Mitsubishi Electric Reseach Laboratories (MERL), Boston (*General Manager*)
1998-: Austin Software Council *(Executive Director, Board member, ...)*
1998-- International consultant in the high tech industry; Advisor or Board member at startup companies (USA, Austria, Hungary);
2003-: Eutecus Inc. (*President*)

Néhány további tevékenység:

1971-72: Visiting Professor, Univ. of California, Berkeley
1974: Visiting Professor, Imperial College, London
1979-83: Editor-in-Chief, IEEE Transaction on Software Engineering
1987-88: Member, USAF Science Advisory Board
1984-94: Chairman, Advisory Board, IT Institute of the National Computer Board of Singapore;
1988-94: Board of Directors, Computer Research Associates

Kitüntetések:

IBM Outstanding Contribution award (1969, 1973)
Citation Index Classic, for the most-referenced paper in the field of software (1983)
IEEE Fellow (1988)
Warnier Award for excellence in information technology (1990)
Award of the John von Neumann Computer Society (2003)

Tudományos munkásság összefoglalása:

- Bélády László tudományos munkásságának jelentős része a számítógépek tároló rendszereinek elméleti és gyakorlati kérdéseihez kapcsolódik. Meghatározó szerepe volt a 60-as évek elején kifejlesztett és azóta a számítógépekben általánosan alkalmazott **virtuális memória** rendszerek alapvető problémáinak megoldásában. Eredményeinek és az általa bevezetett fogalmaknak a hatása napjaink számítógép architektúráiban és operációs rendszereiben is érvényesül. (Részletesebben ld. az - angol nyelvű - 1. mellékletben)

- M. Lehmannal közösen írt “Program Evolution” című könyvét (1985) a programozás elméletének egyik legtöbbet idézett alapműveként tartják számon, amelyben a programok fejlődési dinamikájával kapcsolatos másfél évtizedes kutatások eredményeit foglalják össze.
- Aktív résztvevője a Software Engineering kérdéseivel foglalkozó szakmai közéletnek: több jelentős publikáció mellett programbizottsági elnöke volt ilyen tárgyú IEEE konferenciáknak és négy éven keresztül (1979-83) volt főszerkesztője a szakterület legrangosabb folyóiratának (IEEE Transaction on Software Engineering)
- Az IBM-nél különböző kutatói és kutatás menedzseri beosztásokban eltöltött közel két és fél évtized után két jelentős kutató intézet alapításában és vezetésében játszott fontos szerepet:
    - az információs technológiák fejlesztése érdekében a 80-as évek közepén USA állami kezdeményezésre, a számítástechnikai ipar vezető cégeinek konzorciumaként létrehozott Microelectronics & Computer Technology Corp (MCC) egyik alapító alelnöke és a Software Technology (később System Architecture) projektjének vezetője volt 1984-91 között;
    - a japán Mitsubishi cég által az Egyesült Államokban létesített önálló kutatóintézet (Mitsubishi Electric Reseach Laboratories (MERL) alapító igazgatója volt 1991-98 között.

(Publikációs listát ld. a 2. mellékletben.)

Indoklás:
Bélády László a számítástudomány elméleti és gyakorlati kérdéseivel egyaránt magas színvonalon foglalkozó kutató, aki szakmai tevékenységének nagy részét ipari környezetben végezte. Eredményei jelentősen hozzájárultak a számítógép architektúrák fejlődéséhez valamint a programkészítés folyamatában található törvényszerűségek feltárásához. Közvetlen és közvetett hatásuk több területen ma is érzékelhető.

Világszerte elismert tevékenysége mellett szoros kapcsolatot tart fenn a magyar számítástechnikai szakmával is. Rendszeres látogatásai során igyekszik megismerkedni a hazai kutató-fejlesztő társaságokkal és segíteni nemzetközi kapcsolataik kialakítását. Kiváló kapcsolatot ápol a Műegyetem számos professzorával is.

Bélády László szakmai vezetésével rendezi meg 2002 óta a Neumann János Számítógép-tudományi Társaság a Szoftver Technológiai Fórumot, amelynek általában negyedévenként sorrakerülő rendezvényein általában egy neves külföldi előadó ismertet egy aktuális témát, amit egy vagy több hazai előadás követ. A nagyon sikeres (eddig 23 alkalom volt) rendezvény előadói leggyakrabban Bélády László szerteágazó szakmai kapcsolatai alapján kerülnek meghívásra. (Részletesebben ld. http://www.inf.u-szeged.hu/stf/)

A hazai számítástechnikai szakma érdekében kifejtett tevékenységét ismerte el a Neumann János Számítógép-tudományi Társaság akkor, amikor a Neumann János születésének századik évfordulója alkalmából 2003-ban három olyan magyar tudóst tüntetett ki a Neumann János emlékplakettel, akiknek a tevékenysége jelentős hatással van a világ számítástudományára. Ezek között Lovász László és Simonyi Károly mellett Bélády László is ott volt. Kitüntetésének indoklása: „*A szoftver technológia megalapozása területén elért kimagasló eredményeiért, valamint a nemzetközi trendek magyarországi bemutatásában kifejtett áldozatos munkájáért*”.

Mellékletek:

1. Belady’s technical contributions
2. Publications

1. melléklet

BELADY's TECHNICAL CONTRIBUTIONS

Influence of the Fundamental Contributions

Storage systems are fundamental to computing. Data structure and layout affect algorithm complexity and program completion time. The dynamic movement of information up and down the memory hierarchy has a profound impact on system throughput, response time. and programming productivity.
Today's most influential operating systems -- Windows. MacOS. and Linux all use working set memory management and load balancing. LRU caches provide significant performance enhancements for components and subsystems ranging from CPUs to video cards. to disk Controllers, to databases and networks. to client applications and to the Internet itself. where strategically placed web caches eliminate bottlenecks at popular sites. Locality and working sets have enabled similar advances in related systems including contention managers, high performance computing algorithms and architectures. program restructuring, compilers, databases. networks, graphics, search engines. energy-critical embedded systems, and chip-level caches. The principles of locality and working set have become deeply embodied within the architectures of computer systems and networks. These principles have been so widely successful that they are today virtually transparent and unnoticed. just as CPU architectures are virtually unnoticed. Yet none of these systems would perform well without them. Belady is one of the original discoverers of these principles and the first to develop the analytic tools to exploit them.
Belady helped settle controversies and provided a sound basis for analyzing and designing storage management processes. Rigorous analyses led to one of the first scientific theories in computing: they studied an important phenomenon empirically and modeled it mathematically; their models made non obvious predictions, and these predictions were validated in practice. In addition to enabling accurate performance prediction and stable control of virtual storage systems, their methods were taken into many domains where they again provided the basis for effective performance prediction and control.

Origins

The Atlas virtual memory (late 1950s) was the first system to automate dynamic allocation of programs and data in a memory hierarchy. The Atlas designers invented address space. address mapping. demand paging. and replacement algorithms. They demonstrated that automating storage management could double or triple programmer productivity. Despite the attractive programming benefits, virtual memory was controversial because experimental studies with replacement algorithms led to contradictory results about ultimate system performance.

The controversy came to a head in mid 1960s when major computer makers (RCA. Burroughs, Univac. General Electric, and others) adopted virtual memory. They soon discovered to their utter surprise that multi-programmed virtual memory systems were highly susceptible to thrashing -- the systems would suddenly slow to a barely perceptible crawl and could be recovered only when some jobs were aborted. Engineers of the day called the problem "paging to death." They had no idea why it was happening or how to prevent it. Their new operating systems began to look like multimillion dollar liabilities. Belady began his experiments in program behavior and page replacement algorithms in 1963 while working at IBM Research in Yorktown Heights. His 1966 paper on page replacement algorithms offered an initial definition of locality and demonstrated useful mathematical models of programs executing in storage systems.

Belady developed a simulator for investigating the performance of paging algorithms using actual recorded address traces. He later ran experiments on a multi-programmed virtual storage system he helped design. He discovered MIN, the optimal (look-ahead) paging algorithm broadly used in practice and found in textbooks. lie identified locality as the decisive factor that separated the good replacement algorithms from the failures. He demonstrated that program restructuring could enhance locality and thereby improve performance of replacement algorithms by factors of 2 or 3. He demonstrated that replacement algorithms became even more efficient in multiprogramming environments when the storage partition varied dynamically.

Impacts

Although motivated by problems that first arose in the 1960s, work continued in modeling, analysis and practical application of locality in many systems of the 1970s. It also led to a deep, subtle, broadly applicable, and non obvious scientific theory. On the practical side the work

- Solved the replacement problem, a significant determining factor in the performance of dynamic storage systems. Replacement algorithms decide which information resides in thc top levels of memory. Those that detect locality by observing usage outperform others. Many operating systems exploited this principle, including most recently Windows Vista, MacOS Tiger, and Red Hat Linux.
- provided the analytic tools used subsequently to design caches and caching for CPUs.disk controllers, software applications, web caching, search engines, and forensics
- Laid to rest the myth of flat memory. Flat memory is the idea that memory will become so cheap and large that dynamic storage allocation methods will become obsolete. Locality predicts that some objects are much more popular than others. and the memory will develop performance bottlenecks at those popular sites. Web caching systems are a permanent feature of the Internet.
- Stimulated research and development in many areas of OS, networks, compilers and distributed systems
- Impacted many areas of network design

Finally, as a result of this work, the concepts of locality has become standard in the lexicon and thinking of computer scientists and engineers

Specific contributions

In 1963. he created a tool for capturing the memory reference patterns of real application programs by modifying a software interpreter (written by the late Robert A. Nelson). He then used the captured traces to experiment with alternative page sizes. page replacemenl algorithms and memory capacities. This empirically based work resulted in a number of important theoretical advances that are described in his landmark 1966 paper (see below)

1 "A study of page replacement algorithms for virtual storage computers:' (IBM Systems J 1966). It was quickly recognized for quality, and immediate and lasting impact as in 1983 as the most cited CS software paper of all time.
2. He derived mathematical expressions for the performance of the RAND (random replacement algorithm and validated with empirical data.
3. He observed that the FIFO (first in. first out) replacement algorithm provided superior but sometimes unpredictable performance relative to RAND, and demonstrated empirically that this benefit was due to the tendency of real programs to cluster their references within a subset of their

address spaces. He introduced the term "locality" to characterize this clustering and provided a definition of a "locality set" based on the operation of the FIFO replacement algorithm.

4. He showed empirically that the LRU (least recently used) page replacement algorithm. which also exploits locality. performed better than either FIFO or RAND. He introduced the "used bit" as a practical technique for implementing an approximate LRU page replacement algorithm.
5. He invented the MIN algorithm, which minimized the number of page faults for any page reference trace (a rigorous proof of MIN optimality was later presented in 1971 by Aho. Denning and Ullman). Later, he filed a patent on a device that monitored a CPU's address trace and output the page fault function that MIN would have had, thus showing that optimal paging rate is computable in real time even if memory contents are not! MIN. also referred to as Belady's algorithm, has important applications in other areas including the register allocation phase of compiler operation.
6. Belady's studies of page replacement algorithms were highly influential in the design the M44 computer (a modified IBM 7044 computer that supported multiprogrammed virtual memory). Once the M44 became operational in 1965, he conducted "live" experiments with this new system to gain additional insights into the behavior of computations under dynamic storage partitioning.
7. On the M44 he discovered the "FIFO anomaly" (also called "Belady's anomaly"; which is the observation that increasing the amount of space allocated to a particular program can actually increase the number of page faults under FIFO replacement. The question whether other paging algorithms allow anomalies inspired research into stack algorithms. which led to the discovery that a large class of paging algorithms is free of such anomalies because they satisfy an inclusion property.
8. On the M44 he introduced the lifetime function (mean time between faults as function of memory size) as measure of merit for paging algorithms. He proved that this function has a non-linear shape that begins as a "concave up" curve and then transitions to a "concave down" curve. The inflection point. which he called “parachor” (a term introduced by the architects of the IBM 360). represents a target memory allocation that would achieve reasonable performance.
9. On the M44 he observed that varying partition size improved overall lifetime function. He explained this result mathematically by using the concave up property of individual program lifetime functions.
10. On the M44 he ran experiments with load limits defined by giving individual programs their parachors. With these load controls, the system was much more resistant to "paging itself to death" (an early term for thrashing).
11. He consulted on various IBM experiments that showed how grouping program modules within pages according to "nearness" significantly improved locality and the performance of virtual storage systems. reducing paging 3-10 times for given amount of memory. He consulted on experiments that showed providing simple guidelines to programmers created greater locality in their source (and object) programs.

In addition to the basic work on storage systems, Belady made other notable contributions:

- After completing his research into storage management in 1968. he formed and ran an IBM research team focused on computer graphics. He holds US patents in this area.
- In 1970, he began collaboration with M M Lehman on programming productivity and quality. Their investigations led to an understanding of the way program quality degrades over time because of successive rounds of development and maintenance. They coauthored a highly influential book, Program Evolution. published by Academic Press in 1985.
- In the late 1970s, he led a team within IBM that redesigned a major component of the OS 360 operating system.

Impact on other fields

The locality and working set ideas have been adopted into research and technologies in a wide range of fields. Primary examples are:

- In virtual memory to organize caches for address translation, to design the replacement algorithms, and to manage dynamic load.
- In data caches for CPUs. originally as mainframes and now as microchips.
- In buffers between main memory and secondary memory devices.
- In buffers between computers and networks.
- In video boards to accelerate graphics displays.
- In modules that implement the information-hiding principle.
- In accounting and event logs that monitor activities within a system.
- In alias lists that associate longer names or addresses with short nicknames.
- In the "most recently used" object lists of applications.
- In web browsers to hold recent web pages.
- In file systems, to organize indexes (e.g., B-trees) for fastest retrieval of file blocks.
- In database systems, to manage record-flows between levels of memory. In search engines to find the most relevant responses to queries.
- In classification systems that cluster related data elements into similarity classes. In spam filters, which infer which categories of email are in the user's locality space and which are not.
- In "spread spectrum" video streaming that bypasses network congestion
- and reduces the apparent distance to the video server.
- In "edge servers" to hold recent web pages accessed by anyone in an organization or geographical region.
- In the field of computer forensics to infer criminal motives and intent by correlating event records in many caches.
- In the field of network science by defining hierarchies of self-similar locality structures within complex power-law networks.

2.melléklet

Publications

1. Laszlo A. Belady, ***A Study of Replacement Algorithms for Virtual-Storage Computer,*** IBM Systems Journal, 5(2), pp. 78-101, 1966.
2. Laszlo A. Belady and C. J. Kuehner, ***Dynamic space-sharing in computer systems,*** Commun. ACM, 12(5), pp. 282-288, 1969.
3. Laszlo A. Belady and Robert A. Nelson and Gerald S. Shedler, ***An anomaly in space-time characteristics of certain programs running in a paging machine,*** Commun. ACM, 12(6), pp. 349-353, 1969.
4. Laszlo A. Belady and Mike W. Blasgen and Carlo J. Evangelisti and Robert D. Tennison, ***A Computer Graphics System for Block Diagram Problem,s*** IBM Systems Journal, 10(2), pp. 143-161, 1971
5. Laszlo A. Belady and Frank P. Palermo, ***On-Line Measurement of Paging Behavior by the Multivalued MIN Algorithm,*** IBM Journal of Research and Development, 18(1), pp. 2-19, 1974
6. Laszlo A. Belady and M. M. Lehman, ***A Model of Large Program Development,*** IBM Systems Journal, 15(3), pp. 225-252, 1976.
7. L. A. Belady and M. M. Lehman, ***Characteristics of Large Systems,*** IBM Systems Journal, Number 3, 1977.
8. L. A. Belady and P. M. Merline, ***Evolving Parts and Relations — A Model of System Families,*** IBM, Res.R. No.RC6677, August 1977.
9. Anthony I. Wasserman and Laszlo A. Belady and Philip Enslow, Jr. and C. A. R. Hoare and William A. Wulf, ***Toward the Engineering of Software: Problems of the 80's,*** Proceedings: 3rd International Conference on Software Engineering, p. 341, IEEE Computer Society Press, 1978.
10. Laszlo A. Belady, ***Guest Editorial for the Special Collection from the Third International Conference on Software Engineering,*** IEEE Transactions on Software Engineering, 4(4), pp. 261-262, July 1978.
11. Anthony I. Wasserman and Susan L. Gerhart and Edward F. Miller, Jr. and L. A. Belady and William Waite and William A. Wulf, ***Software engineering: the turning point,*** Computer, 11(9), pp. 30-41, September 1978.
12. Laszlo A. Belady, ***Evolved software for the 80's,*** Computer, 12(2), pp. 79-82, February 1979.
13. L. A. Belady and C. Floyd and I. Stinson and T. Straeter and A. Wasserman, ***Education, Training and Technology Transfer in Software Engineering,*** Proceedings: 4th International Conference on Software Engineering, p. 51, IEEE Computer Society Press, 1979.
14. Laszlo A. Belady, ***Editor's Notice,*** IEEE Transactions on Software Engineering, 6(1), p. 1, January 1980.
15. L. A. Belady and B. Leavenworth, ***Program Modifiability,*** Software Engineering, Academic Press Inc, 1980.
16. Laszlo A. Belady, ***Modifiability of Large Software Systems,*** Proceedings of the 14th IBM Computer Science Symposium on Operating Systems Engineering, LNCS, Vol. 143, pp. 160-174, Springer, October 1980
17. Laszlo A. Belady and Carlo J. Evangelisti and Leigh R. Power, ***GREENPRINT: A Graphic Representation of Structured Programs,*** IBM Systems Journal, 19(4), pp. 542-553, 1980.

18. Laszlo A. Belady and Richard P. Parmelee and Casper A. Scalzi, ***The IBM History of Memory Management Technology,*** IBM Journal of Research and Development, 25(5), pp. 491-504, 1981.
19. Laszlo A. Belady and Carlo J. Evangelisti, ***System partitioning and its measure,*** Journal of Systems and Software, 2(1), pp. 23-29, 1981
20. Laszlo A. Belady, ***Software Parts - Small and Large,*** Software-Qualitätssicherung, Berichte des German Chapter of the ACM, Vol. 9, pp. 270-285, Teubner, 1982.
21. Laszlo A. Belady, ***Editorial***, IEEE Transactions on Software Engineering, 8(4), p. 297, July 1982.
22. Mamoru Maekawa, Laszlo A. Belady, ***Operating Systems Engineering,*** Proceedings of the 14th IBM Computer Science Symposium, Amagi, Japan, Springer 1982
23. Laszlo A. Belady and T. R. N. Rao, ***Editorial,*** IEEE Transactions on Software Engineering, 9(3), p. 217, May 1983.
24. L. A. Belady and K. Hosokawa, ***Visualization of Independence and Dependence for Program Concurrency,*** IEEE Computer Society Workshop on Visual Languages, pp. 59-63, December 1984.
25. L. A. Belady and M. M. Lehman, ***Program evolution: processes of software change,*** Academic Press, 1985.
26. M. M. Lehman and L. A. Belady, ***New Software Engineering Program - Worldwide (Panel),*** ICSE, pp. 128-131, 1985.
27. Laszlo A. Belady, ***Programs, Life Cycles and Laws of Software Evolution,*** Program Evolution: Processes of Software Change, APIC Studies in Data Processing, Vol. 27, Academic Press, 1985.
28. Laszlo A. Belady, ***MCC: Planning the Revolution in Software,*** IEEE Software, 2(6), pp. 68-73, 1985.
29. Laszlo A. Belady and Nico Haberman and Lim Swee Say and John Elmore and W. M. Murray and R. W. Witty and K. Kishida and Fuad Gattaz Sobrinho, ***New software engineering programs—worldwide (panel session)***, Proc. 8th Intnl. Conf. on Software Engineering, 1985.
30. Laszlo A. Belady, ***Software engineer, the system designer,*** ACM Conference on Computer Science, pp. 53-55, ACM, 1986.
31. Laszlo A. Belady, ***The Japanese and Software: Is it a Good Match?*** IEEE Computer, 19(6), pp. 57-61, 1986.
32. Laszlo A. Belady, ***Leonardo: The MCC Software Research Project,*** Software Engineering in the Nineties, XXX, October 1988.
33. James D. Babcock and Laszlo A. Belady and Nancy C. Gore, ***The Evolution of Technology Transfer at MCC's Software Technology Program: From Didactic to Dialectic,*** Proceedings: 12th International Conference on Software Engineering, pp. 290-299, IEEE Computer Society Press / ACM Press, 1990.
34. Laszlo A. Belady, ***From Software Engineering to Knowledge Engineering: The Shape of the Software Industry in the 1990s,*** International Journal of Software Engineering and Knowledge Engineering, 1(1), pp. 1-8, 1991.
35. Laszlo A. Belady, ***Experiments and Measurements for Systems Integration,*** Experimental Software Engineering Issues, Lecture Notes in Computer Science, Vol. 706, pp. 17-18, Springer, 1992.
36. Laszlo A. Belady, ***The Interdisciplinary Future: The (Data) Engineering Point of View Is Not Enough,*** Proc. IEEE CS Intl. Conf. No. 8 on Data Engineering, Tempe, AZ, February 1992.

37. Laszlo A. Belady, ***Boosting the Effectiveness of Industrial Research in Computer Sciences,*** HICSS (4), pp. 473-477, 1994.
38. Laszlo A. Belady, ***The Disappearance of the ``Pure'' Software Industry,*** ACM Computing Surveys, 27(1), pp. 17-18, March 1995.
39. Victor R. Basili and Laszlo Belady and Barry Boehm and Frederick Brooks and James Browne and Richard DeMillo and Stuart I. Feldman and Cordell Green and Butler Lampson and Duncan Lawrie and Nancy Leveson and Nancy Lynch and Mark Weiser and Jeannette Wing, ***NSF workshop on a software research program for the 21st century***, 1999.
40. Laszlo A. Belady, ***Soapbox: Global R&D: How to Break Barriers,*** IEEE Software, 16(4), pp. 15-17, July /August 1999.
41. Laszlo A. Belady, ***Beyond Software and Beyond Engineering,*** CSEE&T, p. 219, 2000. ISESE 2002: 3-6