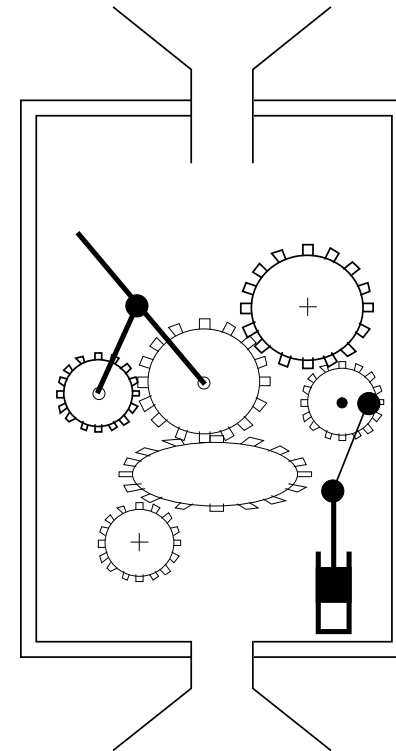


# Die Patent-Story

WRC-Garbage-Collection-Algorithm patentiert:  
Europäische Patentnummer 86309082.5



Ist der resultierende Terminierungs-  
erkennungsalgorithmus auch durch  
das Patent geschützt?

Bekannte Garbage-Collection-Verfahren werden so in bekannte und brauchbare Algorithmen zur Erkennung der verteilten Terminierung transformiert!

# Das WRC-Patent

Europäisches Patentamt  
European Patent Office  
Office européen des brevets

(1) Publication number:  
**0 225 755**  
A2

## (12) EUROPEAN PATENT APPLICATION

(21) Application number: 86309082.5  
(51) Int. Cl.: **G 06 F 1202**

(22) Date of filing: 20.11.86

20.11.86

ICL

(30) Priority: 04.12.85 GB 8529890

(71) Applicant: INTERNATIONAL COMPUTERS LIMITED ICL  
House, Putney, London, SW15 1SW (GB)

Watson, Paul

(43) Date of publication of application: 16.06.87  
Bulletin 87/25

(72) Inventor: Watson, Paul, 146, Hilda Park  
Chester-Le-Siret, Co-Durham DH2 2JY (GB)

(84) Designated Contracting States: BE DE FR GB NL

(74) Representative: Guyatt, Derek Charles et al, STC  
Patents Edinburgh Way Harlow Essex CM20 2SH (GB)

(54) Garbage collection in a computer system

(57) A computer system is described, having memory cells organised in a directed graph structure by means of pointers. Each cell has a reference count, and each pointer a weight value. If a new pointer to a cell is created by copying an existing pointer, the new and existing pointers are given weights whose sum equals the old value of the existing pointer. In this way, the sum of the weights of the pointers to any cell are maintained equal to its reference count. If a pointer is destroyed, the reference count of the cell to which it points is reduced by the weight of the pointer. Thus, when the reference count of a cell reaches zero, it is safe to assume that there are no more existing pointers to it, and hence that cell may be reclaimed (garbage-collected) for re-use.

nächste Folie-->

(84) Designated Contracting States: BE DE FR GB NL

(54) Garbage collection in a computer system

(57) A computer system is described, having **memory cells** organised in a directed graph structure by means of **pointers**. Each **cell** has a **reference count**, and each **pointer** a **weight** value. If a new pointer to a cell is created by **copying** an existing pointer, the new and existing pointers are given weights whose sum equals the old value of the existing pointer. In this way, the **sum of the weights of the pointers to any cell are maintained equal to its reference count**. If a pointer is destroyed, the reference count of the cell to which it points is reduced by the weight of the pointer. Thus, when the reference count of a cell reaches **zero**, it is **safe to assume that there are no more existing pointers to it**, and hence that cell may be **reclaimed** (garbage-collected) for re-use.

Invariante!

# Ein hierarchisches Patent

## CLAIMS:-

1. A *computer system* having storage means containing a plurality of memory cells, at least some of which contain pointers to others of the cells thereby defining a directed graph structure in which each cell represents a node of the graph, wherein

- (a) each *pointer* has a *variable weight value* associated with it,
- (b) each *cell* contains a *variable reference count* value,
- (c) whenever a cell is *allocated* in the storage means, its reference count is initially set to a predetermined value and the weights of any pointers to that cell are set to non-zero values such that the sum of those weights (or the weight if there is only one such pointer) is equal to the reference count of that cell,
- (d) whenever a new pointer to a cell is created by *copying* an existing pointer, the new pointer and the existing pointer are given non-zero weight values whose *sum equals the old weight value* of the existing pointer, but the reference count of the cell is unaltered,
- (e) whenever a pointer is *destroyed*, the reference count of the cell to which it points is reduced by the weight of that pointer, and
- (f) cells with reference *count equal to zero* are reclaimed.

2. A system according to Claim 1 wherein the weight of each pointer is a *power of two*, and wherein, whenever a pointer is created by copying an existing pointer of weight greater than one, the weight of the existing pointer is divided by two and the result is stored as the weight value of the new and existing pointers.

3. A system according to Claim 2 wherein the weight of each pointer is *encoded in a compressed form as the logarithm* to the base two of the weight.

4. A system according to any preceding claim wherein, whenever a pointer is to be created by copying an existing pointer of *weight equal to one*, a *dummy cell* is created into which the existing pointer is copied along with its weight, the dummy cell is given a reference count greater than one, and the new and existing pointers are both made to point to this dummy cell, and are given non-zero weights whose sum equals the reference count of the dummy cell.

5. A system according to any preceding claim in which the system is a *multi-processor distributed computer system*.

# Wo findet man Patente?

- Zum Beispiel beim Deutschen Patentamt: [www.depanet.de](http://www.depanet.de)

- Frühjahr 1999: "Suchkriterien (garbage AND collection)<TITLE>  
179 gefundene(s) Dokument(e)"

- ... bei kommerziellen Information-Brokern

- die verlangen Geld

- ... beim IBM Patent Server: <http://www.patents.ibm.com/>

Search Results

Query: (garbage collection)

139 of 2569032 matched (ein Jahr zuvor: 93 of 2461228).

5652883 Computer method and system for conservative-stack and generational heap garbage collection

5561785 System for allocating and returning storage and collecting garbage using subpool of available blocks

5560003 System and hardware module for incremental real time garbage collection and memory management

5446901 Fault tolerant distributed garbage collection system and method for collecting network objects

5819299 Process for distributed garbage collection

5832529 Methods, apparatus, and product for distributed garbage collection

5033930 Garbage collecting truck

5398334 System for automatic garbage collection using strong and weak encapsulated pointers

4755939 Garbage collection in a computer system

5901540 Garden tool for collection and removal of debris

...

man hätte besser auch nach  
"garbage collector" etc. gefragt

# Beispiele für Softwarepatente (1)

Es gibt viel ganz offensichtliche Dinge, die patentiert wurden, das gilt auch für Algorithmen. Einiges davon ist in der fachlich beschlagenen Öffentlichkeit bekannt geworden, z.B. das LZW-Komprimierungspatent (US4558302) beim gif-Bildformat. Es gibt aber auch noch viele andere "interessante" Patente. Vieldiskutiert ist die Frage, ob Algorithmen (oder ganz allgemein "Intellectual Property") überhaupt patentiert werden soll.

**Any word processor with a separate mode that the user selects when they wish to type in a mathematical formula.** [US5122953]

**A word processor which marks and makes correction to a document using two additional different colors.** [US5021972]

Inventor(s): Nishi; Toshio  
June 4, 1991

*A word processor, including a keyboard through which characters can be inputted, a memory device for storing inputted character arrays and a display device capable of multi-color displays, is so programmed that corrections and additions are automatically displayed in a different color from the rest for the convenience of editing. ... When a completed document is finally stored in a document file, however, such color codes are deleted such that the document can be outputted in one color.*

**Statically allocating an initial amount of memory when a program is first loaded according to a size value contained in the program header.** [US5247674]

Applicant(s): Fujitsu Limited, Kawasaki, Japan  
Issued/Filed Dates: Sept. 21, 1993 / May. 13, 1991

*A memory allocation system includes a unit for storing the information about the amount of memory required at the time of initializing each executable program in the control information of the file storing the program. The amount required is determined when the program is translated, assembled or compiled and linked. The memory allocation system also includes a unit for reading the information, indicating the amount of memory required at the time of initializing the program stored in the control information of the file, when loading of program is requested.*

## Beispiele für Softwarepatente (2)

**Assigning a client request to a server process by first examining all the server processes not handling the maximum number of clients, and then assigning it to the server process currently servicing the fewest clients.** [US5249290]

Applicant(s): AT&T Bell Laboratories, Murray Hill, NJ  
Issued/Filed Dates: Sept. 28, 1993 / Feb. 22, 1991  
*A server of a client/server network uses server processes to access shared server resources in response to service requests from client computers connected to the network. The server uses a measured workload indication to assign a received client service request to a server process... a busy indicator provides a measured workload indication for each active process. The server uses the busy indicator to assign a new client service request to the least busy process.*

**Method for canonical ordering of binary data for portable operating systems.** [US4956809]

Applicant(s): Mark Williams Company, Chicago, IL  
Issued/Filed Dates: Sept. 11, 1990 / Dec. 29, 1988  
*...The method includes converting all binary data accessed from a file or communications channel from the canonical order to the natural order of the host computer before using the binary data in the host computer and converting all binary data which is to be sent to a file or communications channel from the natural order of the host computer to the canonical order before sending the binary data.*

**Remembering file access behavior and using it to control the amount of read-ahead the next time the file is opened.** [US5257370]

**Using of multiple read only tokens and a single read write token to control access to a portion of a file in a distributed file system.** [US5175851]

**Quicksort implemented using a linked list of pointers to the objects to be sorted.** [US5175857]

**Intercepting calls to a network operating system by replacing the first few instructions of an entry point by a call to an intercept routine.** [US5257381]

## Beispiele für Softwarepatente (3)

**Distinguishing nested structures by color.** [US4965765]

Applicant(s): International Business Machines Corp., Armonk, NY  
Issued/Filed Dates: Oct. 23, 1990 / May. 16, 1986  
*A method of distinguishing between nested expressions, functions, logic segments or other text by using a different color for each nesting level.*

**Das berüchtigte "xor-Cursor-Patent": Method for dynamically viewing image elements stored in a random access memory array.** [US4197590]

Issued/Filed Dates: April 8, 1980 / Jan. 19, 1978  
*...An XOR feature allows a selective erase that restores lines crossing or concurrent with erased lines. The XOR feature permits part of the drawing to be moved or "dragged" into place without erasing other parts of the drawing. ...supporting another image on the display without destruction of the initially stored image... logically exclusively ORing together the accessed data for each element of the stored image and the data for the corresponding element of the image to be superimposed, and for reentering the resultant logical data into the same memory locations, said display then being generated from the resultant contents of said memory.*

**A parallelizing compiler that estimates the execution time for each of a number of different parallelization conversions and then selects the one that it thinks will be the fastest.** [US5151991]

**Any document storage system that has a digital camera to scan in documents, stores the documents on an optical disk, and uses character recognition software to construct an index.** [US4941125]

---

Siehe auch:

<http://www.freepatents.org/> (bzgl. der aktuellen Debatte, ob auch in Europa Algorithmen patentiert werden sollen)

<http://www.base.com/software-patents/examples.html> (zu obigen Beispielen)

# Das WRC-Patent (1)

- Die schon genannten Details des Patentes findet man so:

[http://www.patents.ibm.com/details?patent\\_number=4755939](http://www.patents.ibm.com/details?patent_number=4755939)  
4755939 : Garbage collection in a computer system  
INVENTORS: Watson; Paul, Chester-Le-Street, Great Britain  
...



**4755939 : Garbage collection in a computer system**

INVENTORS: **Watson; Paul**, Chester-Le-Street, Great Britain  
ASSIGNEES: **International Computers Limited**, London, England  
ISSUED: **July 5, 1988** FILED: **Nov. 24, 1986**  
SERIAL NUMBER: **934439** MAINT. STATUS:  
INTL. CLASS (Ed. 4): **G06F 011/00**;  
U.S. CLASS: **364/300**  
FIELD OF SEARCH: **364-200 MS File,900 MS File,300 ;**

**ABSTRACT:** A computer system is described, having memory cells organized in a directed graph structure by means of pointers. Each cell has a reference count, and each pointer has a weight value. If a new pointer to a cell is created by copying an existing pointer, the new and existing pointers are given weights whose sum equals the old value of the existing pointer. In this way, the sum of the weights of the pointers to any cell are maintained equal to its reference count. If a pointer is destroyed, the reference count of the cell to which it points is reduced by the weight of the pointer. Thus, when the reference count of a cell reaches zero, it is safe to assume that there are no more existing pointers to it, and hence that cell may be reclaimed (garbage-collected) for re-use.

# Das WRC-Patent (2)

U.S. REFERENCES: [Show the 19 patents that reference this one](#)

| Patent                  | Inventor      | Issued  | Title   |
|-------------------------|---------------|---------|---|
| <a href="#">4121286</a> | Venton et al. | 10/1978 | <a href="#">Data processing memory space allocation and deallocation arrangements</a> |
| <a href="#">4325120</a> | Colley et al. | 4/1982  | <a href="#">Data processing system</a>  |

EXEMPLARY CLAIM(s): [Show all 10 claims](#)

I claim:

1. A method of garbage collection in a computer system having storage means containing a plurality of memory cells, at least some of which contain pointers to others of the cells thereby defining a directed graph structure in which each cell represents a node of the graph, wherein the method comprises steps as follows:

- (a) each pointer is provided with a variable weight value,
- (b) each cell is provided with a variable reference count value,
- (c) whenever a cell is allocated in the storage means, the reference count of the cell is initially set to a predetermined value and the weights of any pointers to that cell are set to non-zero values such that the sum of those weights (or the weight if there is only one such pointer) is equal to the reference count of that cell,
- (d) whenever a new pointer to a cell is created by copying an existing pointer, the new pointer and the existing pointer are given non-zero weight values whose sum equals the old weight value of the existing pointer, but the reference count of the cell is unaltered,
- (e) whenever a pointer is destroyed, the reference count of the cell to which the pointer points is reduced by the weight of that pointer,
- (f) and cells with reference count equal to zero are reclaimed.

RELATED U.S. APPLICATIONS: **none**

FOREIGN APPLICATION PRIORITY DATA:

| Application No. | Country        | Date         |
|-----------------|----------------|--------------|
| 8529890         | United Kingdom | Dec. 4, 1985 |

FOREIGN REFERENCES: **none**

OTHER REFERENCES:

- Alice-A Multi-Processor Reduction Machine for the Parallel Evaluation of Applicative Languages, Darlington et al., Proceedings of the ACM Conference on Functional Programming Languages and Computer Architecture, Oct., 1981.

ATTORNEY, AGENT, or FIRM: **Lee & Smith;**

PRIMARY/ASSISTANT EXAMINERS: **Heckler; Thomas M.;**

## 10 CLAIMS

hier wird nun eine Methode patentiert, nicht mehr wie im europäischen Patent, ein Computersystem!

I claim:

1. A method of garbage collection in a computer system having storage means containing a plurality of memory cells, at least some of which contain pointers to others of the cells thereby defining a directed graph structure in which each cell represents a node of the graph, wherein the method comprises steps as follows:
  - (a) each pointer is provided with a variable weight value,
  - (b) each cell is provided with a variable reference count value,
  - (c) whenever a cell is allocated in the storage means, the reference count of the cell is initially set to a predetermined value and the weights of any pointers to that cell are set to non-zero values such that the sum of those weights (or the weight if there is only one such pointer) is equal to the reference count of that cell,
  - (d) whenever a new pointer to a cell is created by copying an existing pointer, the new pointer and the existing pointer are given non-zero weight values whose sum equals the old weight value of the existing pointer, but the reference count of the cell is unaltered,
  - (e) whenever a pointer is destroyed, the reference count of the cell to which the pointer points is reduced by the weight of that pointer,
  - (f) and cells with reference count equal to zero are reclaimed.
2. A method according to claim 1 wherein the weight of each pointer is a power of two, and wherein, whenever a pointer is created by copying an existing pointer of weight greater than one, the weight of the existing pointer is divided by two and the result is stored as the weight value of the new and existing pointers.
3. A method according to claim 2 wherein the weight of each pointer is encoded in a compressed form as the logarithm to the base two of the weight.
4. A method according to claim 3 wherein, whenever a pointer is to be created by copying an existing pointer of weight equal to one, a dummy cell is created into which the existing pointer is copied along with its weight, the dummy cell is given a reference count greater than one, and the new and existing pointers are both made to point to this dummy cell, and are given non-zero weights whose sum equals the reference count of the dummy cell.
5. A method according to claim 3 in which the system is a multi-processor distributed computer system.
6. A method according to claim 2 wherein, whenever a pointer is to be created by copying an existing pointer of weight equal to one, a dummy cell is created into which the existing pointer is copied along with its weight, the dummy cell is given a reference count greater than one, and the new and existing pointers are both made to point to this dummy cell, and are given non-zero weights whose sum equals the reference count of the dummy cell.
7. A method according to claim 2 in which the system is a multi-processor distributed computer system.
8. A method according to claim 1 wherein, whenever a pointer is to be created by copying an existing pointer of weight equal to one, a dummy cell is created into which the existing pointer is copied along with its weight, the dummy cell is given a reference count greater than one, and the new and existing pointers are both made to point to this dummy cell, and are given non-zero weights whose sum equals the reference count of the dummy cell.
9. A method according to claim 8 in which the computer system is a multiple-processor distributed computer system.
10. A method according to claim 1 in which the computer system is a multi-processor distributed computer system.

## Andere Garbage-Collection-Patente

- Es gibt viele Patente zum Thema "Garbage Collection", und manches hätte man vielleicht selbst erfinden können
  - Titel: Fault tolerant distributed garbage collection system and method for collecting network objects
  - Veröffentlichungsnr.: US5446901
  - Veröffentlichungsdatum: 1995-08-29
  - Erfinder: OWICKI SUSAN S (US); BIRRELL ANDREW D (US); NELSON CHARLES G (US); WOBBER EDWARD P (US)
  - Anmelder: DIGITAL EQUIPMENT CORP (US)
  - Klassifikationssymbol (IPC): G06F12/00

A distributed computer system includes a multiplicity of concurrently active processes. Each object is owned by one process. Objects are accessible to processes other than the object's owner. Each process, when it receives a handle to an object owned by any other process, sends a first "dirty" message to the object's owner indicating that the object is in use. When a process permanently ceases use of an object handle, it sends a second "clean" message to the object's owner indicating that the object is no longer in use. Each object's owner receives the first and second messages concerning usage of that object, stores data for keeping track of which other processes have a handle to that object and sends acknowledgement messages in return. The receiver of an object handle does not use the handle until its first message is acknowledged. Periodically, the object's owner sends status request messages to other processes with outstanding handles to that object to determine if any of those processes have terminated and updates its stored object usage data accordingly. A garbage collection process collects objects for which the usage data indicates that no process has a handle. The first and second messages include sequence numbers, wherein the sequence numbers sent by any process change in value monotonically in accordance with when the message is sent. Object owners ignore any message whose sequence number indicates that it was sent earlier than another message for the same object that previously received from the same process.

## Andere Garbage-Collection-Patente (2)

- Man findet auch Querverweise auf viele andere Patente, auf wissenschaftliche Literatur dazu etc., z.B.:

5355483 : Asynchronous garbage collection  
INVENTORS: Serlet; Bertrand, Paris, France  
ASSIGNEES: NeXT Computers, Redwood City, CA  
ISSUED: Oct. 11, 1994  
FILED: July 18, 1991  
SERIAL NUMBER: 732453

...With this method, the process being collected communicates its memory state ("a memory snapshot") to a garbage collecting process (GC), and the GC process scans the memory and sends back the information about garbage. As a result, the present invention permits garbage collection to be performed asynchronously. The process being scanned for garbage is interrupted only briefly, to obtain the memory snapshot. The process then runs without interruption while the garbage collection is being performed. The present invention makes the assumption that if an object is garbage at the time of the memory snapshot it remains garbage any time later,...

- 
- Es gibt auch noch andere Patent-Server im WWW

- z.B.: <http://metalab.unc.edu/patents/intropat.html>
- patent titles retrieval using the class/subclass code
- the US Patent and Trademark Office uses a classification scheme with 400+ main classes, and tens of thousands of subclasses, to classify all of the 5,300,000+ patents issued since the 1800's.
- man versuche es mit Klasse 364 ("Electrical Computers and Data Processing Systems") und der Subklasse 281.1 ("Memory reclamation/garbage collection")

## Bemerkungen zum WRC-Patent...

From: munnari!tasis.utas.oz.au!ben@uunet.UU.NET (Ben Lian)  
Subject: Watson & Watson's Weighted Reference Counting  
Date: 19 Jun 89 11:55:52 GMT

... I thought you'd like to know that the Watson & Watson algorithm was PATENTED (yes, patented) by ICL in the early '80s. According to one of my colleagues, **the patent covers all uses of the algorithm in which the reference weight is split** (in some arbitrary ratio). This suggests that the patent covers most ANY application of the algorithm. Bevan's algorithm, also in the PARLE 87 proceedings, is an elegant subset of Watson & Watson's. (Ironically, Bevan won the 'best paper' award at the conference!) **The patent permits the use of the algorithm in research contexts only; use in a commercial product is not allowed.** (Which means that our Department's VLSI parallel combinator reduction machine will never see the light of day.)

Ben Lian

P.S. On hearing that the algorithm had been patented, my colleague ran up and down the corridor of our hut, exclaiming: **"How could they do this? They might as well patent the addition operator!"**

---

|                        |  |
|------------------------|--|
| Benjamin Y H Lian      | ACSnet: ben@tasis.utas.oz                  |
| Dept. of EE & CS       | ARPA : ben%tasis.utas.oz.au@uunet.uu.net   |
| University of Tasmania | BITnet: munnari!tasis.utas.oz!ben@         |
| GPO Box 252C           | uunet.uu.net                               |
| Hobart, Tasmania 7001  | UUCP : {eneah, hplabs, mcvax, uunet, ukc}! |
| A U S T R A L I A      | munari!tasis.utas.oz!ben                   |



## Ein weiterer Brief von Ben Lian

As I see it, since the algorithm itself is patented, then **any technique which makes use of weighted reference counts in any form is covered by the patent. This is regardless of different terminology** and/or the strategy used to split reference weights.

I hope this has been helpful. If you have any further questions you might perhaps like to write to Andrew Partridge in my Department. He will be able to provide you with more accurate information—I am only an interested bystander! Alternatively, you can write to Derek Guyatt or Paul Watson (as per the enclosed letter from the latter).

Yours sincerely



Ben Lian  
(ben@tasis.uucp)

## Watson war nicht der erste...

From: grit@carlos.llnl.gov (Dale Grit)  
Message-Id: <8907101526.AA12531@carlos.llnl.gov>  
To: ben@tasis.utas.oz  
Subject: weighted reference counting  
Status: R

I had heard about weighted reference counts in 1980 at a workshop at MIT, so **I asked Arvind** what he knew about them. He responded with the following:

"On reference weights: I brought the idea to the attention of Bob Thomas while he was doing his thesis at Irvine. I had heard it from Ken Weng in course of our many informal discussions. It is explained in a foot note in Ken Weng's Ph.D. thesis. **I have mentioned the idea in my dataflow course at MIT as early as 1980.** Paul (not our old buddy Ian) Watson gave a talk about it at MIT in 1986 (I think). He was quite surprised to learn that we knew about it so thoroughly. He acknowledges it in his Parle paper. He for a while was erroneously attributing the idea to me but I had him correct it. I did not know about the ICL patent. I will be surprised if it was filed in 1982. **In any case I would not worry about it because it can't hold up in court if challenged.** In fact 70 to 80% patents don't hold up when challenged."

If you need more information, please contact Arvind at  
arvind@au-bon-pain.lcs.mit.edu

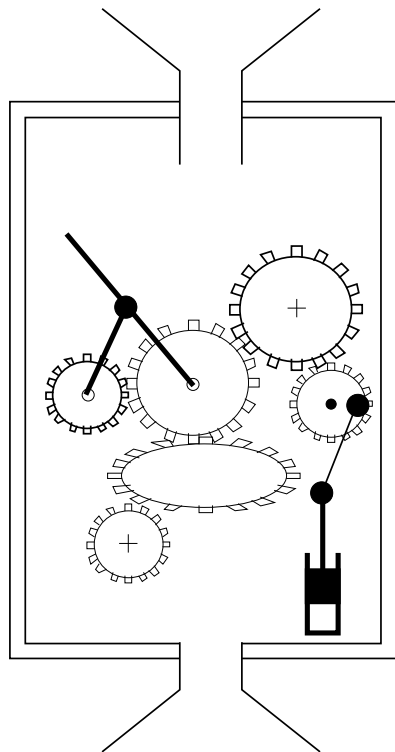
Dale Grit,  
Computer Science Dept.,  
Colorado State University  
on leave at LLNL  
grit@lll-crg.llnl.gov

# Local reference counting (LRC)

**Y. Ichisugi, A. Yonezawa:** Distributed Garbage Collection Using Group Reference Counting, TR 90-014, Univ. of Tokyo

**M. Rudalics:** Implementation of Distributed Reference Counts, Internal Report, J. Kepler University, Linz

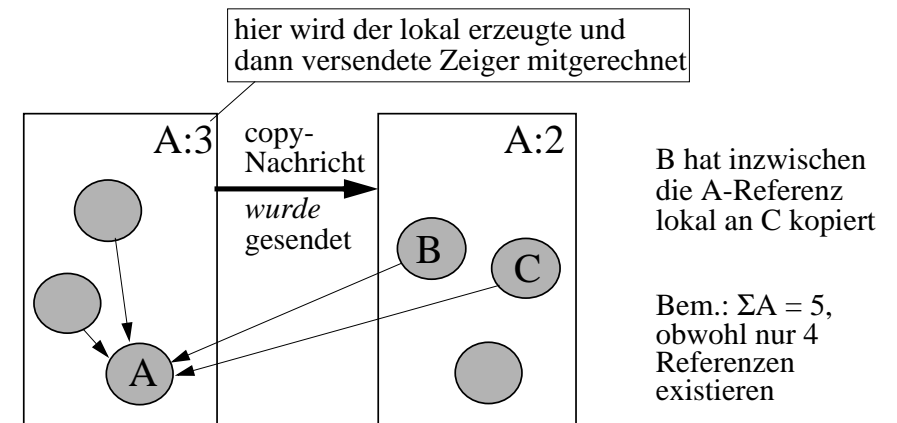
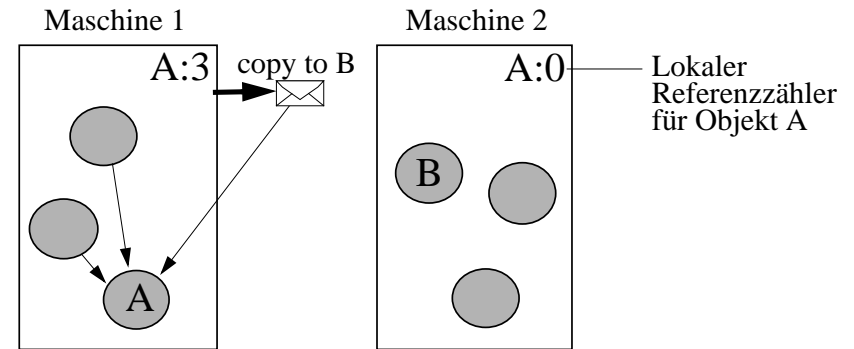
**J. Piquer:** Indirect Reference Counting, Proc. PARLE 91, 150-165




**E.W. Dijkstra, C.S. Scholten:** Termination Detection for Diffusing Computations. Inf. Proc. Lett. 11 (1980), 1-4

# LRC

- Pro Maschine wird (potentiell) für jedes Objekt ein Referenzzähler gehalten, z.B. für Objekt A:



 ← Dec-Nachricht senden, wenn der lokale Referenzzähler bei Erhalt einer dec-Nachricht 0 wird (an den ursprünglichen Sender!)

Beachte: Es wird angenommen, dass der lokale Referenzzähler *atomar* mit den Operationen copy, delete, Empfang einer dec-Nachricht aktualisiert werden kann