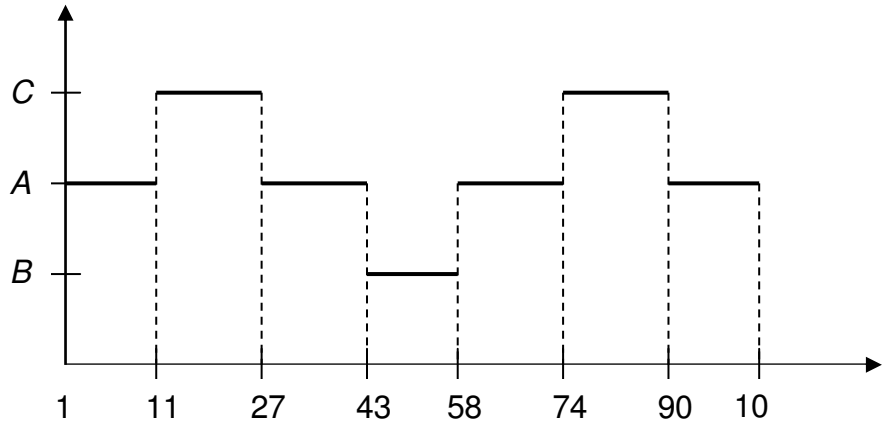
Oblikovanje ispitnih primjera

Za sve promatrane vrste okruženja, neovisno o mjerilu vrednovanja, definirani su ispitni primjeri uz pomoć kojih se ocjenjuje učinkovitost izvedenih pravila raspoređivanja. Ispitni primjeri su podijeljeni u dvije skupine: skup primjera koji se koriste u procesu učenja i skup primjera koji se koriste samo za ocjenu. Način stvaranja ispitnih primjera u skladu je sa pristupom koji se koristi u literaturi, a posebno je opisan za svako okruženje. Skupini primjera za ocjenu dodani su postojeći raspoloživi primjeri iz literature (dobavljeni preko WWW stranica autora), ukoliko su dostupni za određeno okruženje.

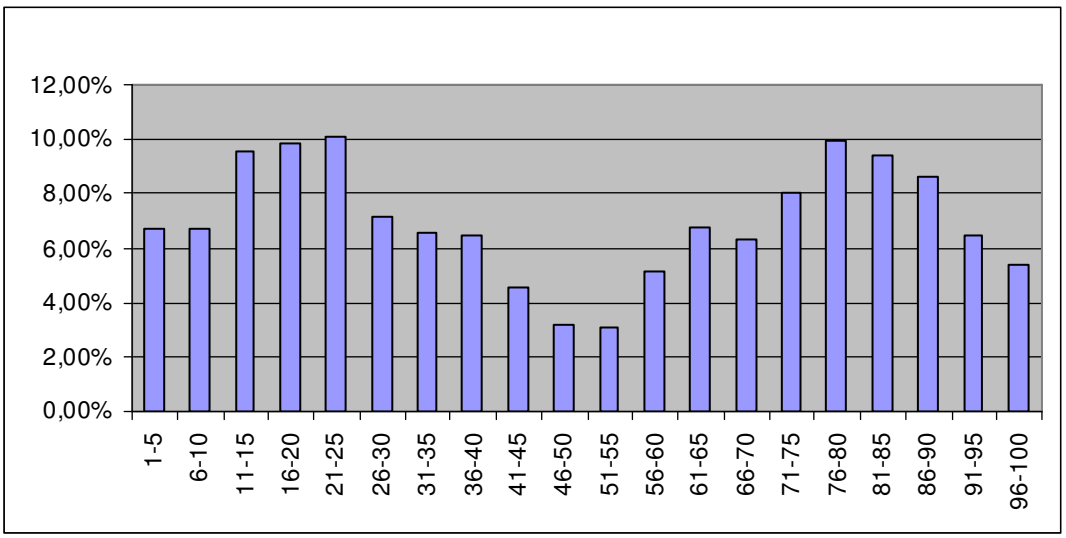
Raspodjele slučajnih varijabli

U postupku definiranja ispitnih primjera nužno je koristiti generatore pseudoslučajnih brojeva koji služe kao osnova za trajanje poslova, vremena dolaska, trajanja postavljanja itd. Pri tome je potrebno odabrati i vjerojatnosnu raspodjelu koju će generirane slučajne varijable slijediti. U skupovima za učenje koristi se jednolika raspodjela slučajnih varijabli dobivena generatorom ugrađenom u MS Visual Studio okolinu. Za potrebe skupa za ocjenu, kombinirane su tri raspodjele: jednolika, normalna (Gaussova) raspodjela i kvazi-bimodalna raspodjela (po uzoru na [Gre 01]). Prilikom sastavljanja skupa za ocjenu navedene raspodjele su korištene u sljedećim udjelima: 20% slučajnih vrijednosti slijedi jednoliku, 50% normalnu a 30% kvazi-bimodalnu raspodjelu. Vjerojatnosna funkcija za kvazi-bimodalnu raspodjelu definirana je kao na slici 1 uz vrijednosti konstanti $B = 0.004651$, $A = 2B$ i $C = 3B$ (vrijednosti slučajnih varijabli postavljene su u intervalu [1, 100]).



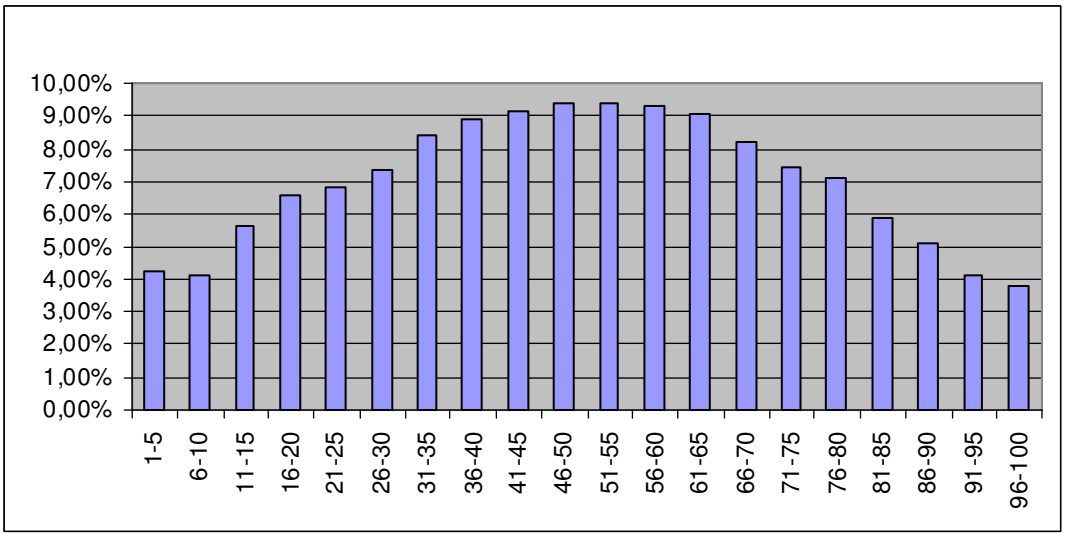
Slika 1. Vjerojatnosna funkcija kvazi-bimodalne raspodjele

Primjer skupa vrijednosti slučajnih varijabli dobivenih kvazi-bimodalnom raspodjelom prikazan je na slici 2.



Slika 2. Primjer skupa vrijednosti dobivenih bimodalnom raspodjelom

Kumulativna raspodjela dobivena kombinacijom prethodne tri prikazana je na slici 3.



Slika 3. Primjer skupa vrijednosti dobivenih kombinacijom jednolike, normalne i bimodalne raspodjele

Odabir najboljeg rješenja

U svakom izvedenom pokusu u pojedinom okruženju promatrana je konačna funkcija dobrote najboljeg rješenja na kraju rada genetskog programa. Prilikom odabira najboljeg rješenja za sve provedene pokuse, najbolja rješenja iz pojedinih pokusa ocijenjena su uz pomoć skupa ispitnih primjera za ocjenu, tj. na onom skupu koji im nije bio na raspolaganju prilikom učenja. Najbolje rješenje za sve pokuse je tada definirano kao ono rješenje koje je postiglo najbolji rezultat na skupu za ocjenu. Smisao ovakvog postupka je biranje onog rješenja (tj. pravila raspoređivanja) koje se najbolje ponaša na skupovima koje prije 'nije vidjelo', odnosno pronalazak onog rješenja koje najbolje poopćava znanje iz primjera za učenje.

Prilikom odabira najboljeg algoritma također je promatran i broj ispitnih primjera u kojima je određeni algoritam postigao rezultat koji ili nije lošiji od rezultata bilo kojeg drugog

promatranog algoritma ili predstavlja najbolji pronađeni rezultat. Ova se mjera može nazvati i *postotkom dominacije*, a obično je povezana sa ukupnom vrijednošću koju algoritam postiže za pojedino mjerilo u svim ispitnim primjerima. Postotak dominacije može doprinijeti izboru jednoga algoritma ukoliko se promatrani algoritmi vrlo malo razlikuju po pitanju ukupne vrijednosti rezultata, što je obično slučaj sa rješenjima dobivenim u kasnijim stupnjevima evolucijskog procesa.

1. Okruženje jednoga stroja

Ispitni primjeri za statičku okolinu

Ispitni primjeri za statičku okolinu definiraju se sljedećim elementima: trajanjima poslova, težinama poslova i željenim vremenima završetka. Trajanje svakog posla može poprimiti cjelobrojne vrijednosti u intervalu od 1 do 100, a težine poslova vrijednosti od 0.01 do 1 u koracima od 0.01. Raspodjele slučajnih varijabli za trajanje poslova opisane su u prethodnom odjeljku, a sve ostale veličine dobivene su po jednolikoj raspodjeli. Za svaki ispitni primjer su, osim toga, definirana dva dodatna parametra s pomoću kojih se računaju željena vremena završetka poslova. Parametar T je postotak zaostajanja (engl. *due date tightness*), a parametar R je područje zaostajanja (engl. *due date range*) [Lee 97]; oba parametra poprimaju vrijednosti u intervalu $[0,1]$. Za svaki ispitni primjer željena vremena završetka definiraju se jednolikom raspodjelom unutar intervala

$$d_j \in \left[\sum_{j=1}^{n} p_j \left(1-T-R/2\right), \sum_{j=1}^{n} p_j \left(1-T+R/2\right) \right], \qquad (1.1)$$

gdje n predstavlja broj poslova u ispitnom primjeru, uz ograničenje da dobivena vrijednost ne može biti manja od nule. Značenje ovih dvaju parametara je sljedeće: parametrom T zadajemo očekivani postotak zakašnjelih poslova, dok parametrom R određujemo širinu intervala vrijednosti koje željena vremena završetka mogu poprimiti. Na primjer, uz $T=1$ očekuje se da će svi poslovi biti zaostali, premda takva situacija nije previše realna.

Za potrebe ocjenjivanja definirano je 100 ispitnih primjera za učenje i 600 ispitnih primjera za naknadnu ocjenu učinkovitosti pravila (genetskom programu su na raspolaganju prvih 100 primjera, dok se kvaliteta izvedenog pravila ocjenjuje na temelju drugih 600 primjera koji nisu dostupni tijekom učenja). Za svaki ispitni primjer određen je broj poslova i vrijednosti parametara T i R. Broj poslova u ovom okruženju poprima vrijednosti od 12, 25, 50 i 100, a parametri T i R poprimaju vrijednosti 0.2, 0.4, 0.6, 0.8 i 1 u različitim kombinacijama. Osim toga, iz raznih izvora [Bea 90] preuzeto je 375 dodatnih ispitnih primjera koji se koriste samo za ocjenjivanje.

Ispitni primjeri za dinamičku okolinu

Ispitni primjeri definirani su na sličan način kao i za statičku okolinu: pojedini primjer opisan je različitim vrijednostima parametara T i R, a područja dopuštenih vrijednosti jednaka su onima za statičku okolinu. Razlika u odnosu na prethodnu okolinu je uvođenje vremena pripravnosti poslova i različito računanje željenog vremena završetka. Za svaki ispitni primjer prvo su određena trajanja poslova te je izračunato ukupno trajanje svih poslova. Vremena pripravnosti generirana su jednolikom raspodjelom u intervalu

$$r_j \in \left[0, \frac{1}{2} \sum_{i=1}^{n} p_i \right]. \qquad (1.2)$$

Željeno vrijeme završetka pojedinog posla definirano je jednolikom raspodjelom u intervalu

$$d_j \in \left[r_j + \left(\sum_{j=1}^{n} p_j - r_j \right) \cdot (1 - T - R/2),\ r_j + \left(\sum_{j=1}^{n} p_j - r_j \right) \cdot (1 - T + R/2) \right]. \qquad (1.3)$$

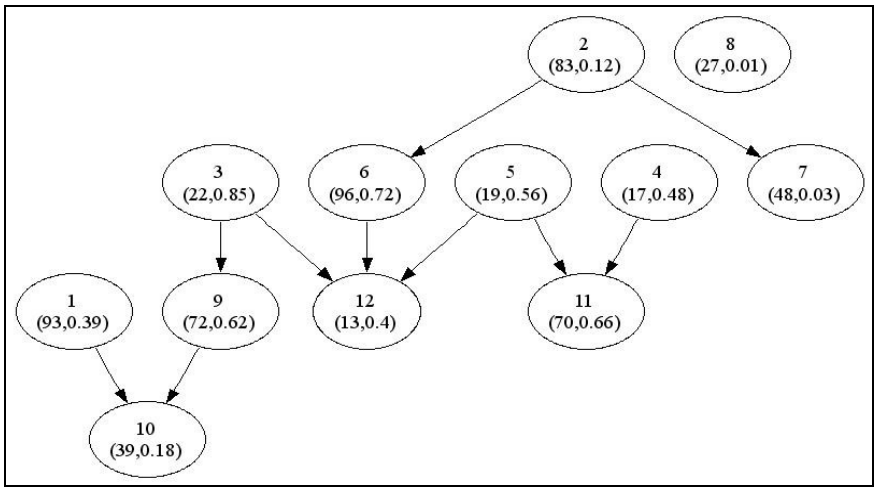
Razlika u odnosu na određivanje željenih vremena završetka u statičkoj okolini (po izrazu (1.1)) je u tome da se za srednju vrijednost uzima razlika ukupnog trajanja i vremena pripravnosti posla, a dobivena vrijednost zbraja se sa vremenom pripravnosti. Kao i u prethodnom odjeljku, načinjeno je 100 ispitnih primjera za učenje i 600 primjera za ocjenu.

Ispitni primjeri uz ograničenja u redoslijedu

Stupanj očekivane zakašnjelosti poslova u ovoj je okolini definiran kao i u prethodnim primjerima s pomoću niza vrijednosti parametara T i R za svaki ispitni primjer. Osim toga, za svaki primjer dodana je i informacija o zadanim međuovisnostima poslova. Ograničenja su u ispitnim primjerima oblikovana u najopćenitijem obliku (ne poprimaju oblik lanaca ili jedinstvenog stabla i sl.), što je ujedno i najteži oblik problema. Prilikom stvaranja ograničenja rabljeni su sljedeći parametri:

- prosječan udio poslova koji će imati prethodnike je 80%,
- udio poslova bez prethodnika nije manji od 20% ,
- najveći broj neposrednih prethodnika posla je 3,
- najveći broj neposrednih sljedbenika posla je 4.

Pokusi su obavljeni i sa različitim vrijednostima od navedenih, no relativni odnosi po pitanju učinkovitosti u tim slučajevima su gotovo nepromijenjeni. Ograničenja su predstavljena u obliku grafova koji su za sve ispitne slučajeve zapisani u obliku pogodnom za korištenje prilikom ispitivanja pravila raspoređivanja. Primjer grafa ovisnosti za ispitni primjer sa 12 poslova prikazan je na slici 4.



Slika 4. Prikaz grafa ovisnosti u ispitnom primjeru s 12 poslova

U gornjem primjeru u svaki čvor grafa upisan je redni broj posla, a u zagradi trajanje posla i njegova težina. Slično kao i u prethodnim problemima, definirano je 100 ispitnih primjera za učenje i 600 primjera za ocjenu.

Ispitni primjeri uz trajanja postavljanja

Prilikom definiranja ispitnih primjera također su rabljeni parametri T i R za određivanje stupnja i područja zakašnjelosti poslova kao u izrazu (1.1). Pored tih podataka, potrebno je generirati trajanja postavljanja za svaku moguću kombinaciju prethodnog i

trenutnog posla, što za jedan ispitni primjer čini matricu veličine $n \times n$. U određenim se situacijama može pretpostaviti da je trajanje postavljanja između dva posla neovisno o njihovom poretku, tj. da je matrica trajanja simetrična, no ovdje će se promatrati općeniti slučaj.

Količina podataka koja bi bila potrebna za zapisivanje trajanja postavljanja za svaki ispitni primjer je neprikladno velika (npr. za 600 primjera za ocjenu, uz prosječno 50 poslova, bilo bi potrebno zapisati 50 x 50 x 600 različitih vrijednosti). Zbog toga se za svaki ispitni primjer trajanja postavljanja definiraju tijekom rada programa, ali tako da se uvijek dobivaju jednake vrijednosti (radi poredbe različitih metoda). U programskom ostvarenju koristi se ugrađeni generator slučajnih brojeva na način da se za svaki ispitni primjer početna vrijednost generatora postavi na redni broj dotičnog ispitnog primjera. Trajanja postavljanja određuju se po jednolikoj raspodjeli uz dodatni parametar η koji predstavlja omjer prosječnog trajanja postavljanja i prosječnog trajanja obrade posla. Za potrebe ispitivanja vrijednost toga parametra postavljena je na 0.5 u primjerima za učenje te 0.5 i 1 u primjerima za ocjenu.

2. Okruženje paralelnih jednolikih strojeva

Tvorba ispitnih primjera za okruženje jednolikih strojeva temelji se na postupcima prikazanim u prethodnom odjeljku, no uz neke dodatne elemente. Broj poslova u ispitnim primjerima za učenje može biti 12, 25, 50 i 100, a u primjerima za ocjenu 25, 50 i 100. Nominalne vrijednosti trajanja poslova dobivaju se opisanim raspodjelama kao cjelobrojne vrijednosti u opsegu od 1 do 100. Težine poslova također poprimaju vrijednosti od 0.01 do 1 u koracima od 0.01. Za svaki ispitni primjer definira se i broj strojeva, koji za primjere za učenje poprima vrijednosti 3, 6 i 10, a u primjerima za ocjenu 3, 6, 10, 15 i 20.

Pored nominalnog trajanja poslova, potrebno je za svaki stroj *i* definirati njegovu brzinu s_i. Trajanje obrade posla j na stroju *i* iznosi

$$p_{ij} = p_j / s_i \,. \tag{1.4}$$

Brzine strojeva se za sve ispitne primjere definiraju na jednak način: definira se slučajna varijabla *spd* koja za svaki stroj poprima vrijednosti u intervalu $[0.1,1]$ u koracima od 0.01 po jednolikoj raspodjeli. Brzina stroja se tada računa kao:

$$s_i = \frac{1}{spd} \tag{1.5}$$

Ovako definirane brzine strojeva poprimaju vrijednosti od 1 do 10, ali uz veće grupiranje kod manjih vrijednosti brzina. Na temelju brzina strojeva moguće je definirati *efektivni broj strojeva* $\hat{m}$ kao zbroj svih njihovih brzina:

$$\hat{m} = \sum_{i=1}^{m} s_i \,, \tag{1.6}$$

gdje je m broj strojeva u dotičnom ispitnom primjeru. Efektivni broj strojeva može se smatrati brzinom jednoga sredstva koje zamjenjuje sve strojeve u sustavu. Uz pomoć efektivnog broja strojeva definiramo očekivano ukupno trajanje poslova $\hat{p}$ kao

$$\hat{p} = \frac{1}{\hat{m}} \sum_{j=1}^{n} p_j \tag{1.7}$$

Vremena pripravnosti poslova generiraju se po jednolikoj raspodjeli u intervalu

$$r_j \in \left[0, \frac{\hat{p}}{2}\right] \tag{1.8}$$

Za definiranje željenih vremena završetaka također se koriste parametri T i R, no umjesto zbroja nominalnih trajanja poslova, koristi se očekivano ukupno trajanje poslova, pa u statičkoj okolini željena vremena završetka poprimaju vrijednosti u intervalu

$$d_j \in \left[\hat{p}\left(1-T-R/2\right), \hat{p}\left(1-T+R/2\right)\right], \tag{1.9}$$

a u dinamičkoj okolini u intervalu

$$d_j \in \left[r_j + \left(\hat{p}-r_j\right)\cdot\left(1-T-R/2\right), r_j + \left(\hat{p}-r_j\right)\cdot\left(1-T+R/2\right)\right]. \tag{1.10}$$

Eventualna trajanja postavljanja poslova definiraju se na način jednak opisanom u prethodnom odjeljku i ne ovise o brzini stroja na kojemu se postavljanje odvija.

3. Okruženje paralelnih nesrodnih strojeva

Ispitni primjeri se u ovome okruženju definiraju slično kao i za jednolike strojeve, uz sljedeće razlike:

- za svaki posao slučajno se definira trajanje izvođenja posla na svakom stroju (p_{ij}), što čini $n \times m$ vrijednosti za svaki ispitni primjer, gdje je n broj poslova a m broj strojeva u dotičnom primjeru;
- očekivano ukupno trajanje poslova $\hat{p}$ definira se kao

$$\hat{p} = \frac{\sum_{j=1}^{n}\sum_{i=1}^{m} p_{ij}}{m^2}. \tag{1.11}$$

U nazivniku gornjeg izraza nalazi se kvadrat broja strojeva jer promatrani algoritmi u većini primjera poslove uspijevaju rasporediti na strojeve koji ih izvode u trajanju manjem od prosječnog trajanja izvođenja posla. Uz pomoć očekivanog ukupnog trajanja i parametara T i R, dolasci poslova generiraju se po izrazu (1.8), a željeno vrijeme završetka poslova računa se po izrazu (1.10). Definirano je 120 primjera za učenje i 600 primjera za ocjenu. Funkcija cilja oblikuje se na jednak način kao i za paralelne jednolike strojeve uz srednje trajanje izvođenja izraženo sa $\overline{p} = \hat{p}/n$.

4. Okruženje proizvoljne obrade

U okruženju proizvoljne obrade potrebno je za svaki posao definirati trajanje svake operacije i njihov redoslijed po strojevima. Trajanje pojedinih operacija generira se na način jednak oblikovanju trajanja poslova u ostalim okruženjima, odnosno po definiranim raspodjelama u intervalu od 1 do 100. Redoslijed operacija dobiva se tako da se na početku definira niz rednih brojeva strojeva, poredan po veličini. Potom se redno svaki element niza zamjenjuje sa nekim drugim slučajno odabranim elementom, nakon čega dobiveni niz predstavlja raspored operacija pojedinog posla. Očekivano ukupno trajanje poslova se za pojedini ispitni primjer definira kao:

$$\hat{p} = \frac{1}{m} \sum_{j=1}^{n} \sum_{i=1}^{m} p_{ij} , \tag{1.12}$$

gdje je n broj poslova a m broj strojeva u dotičnom primjeru. Željena vremena završetka poslova definiraju se uz pomoć opisanih parametara T i R za jednolike paralelne strojeve, po izrazima (1.9) i (1.10).

Svi ispitni primjeri podijeljeni su u skup od 160 primjera za učenje i skup od 320 primjera za ocjenu. Osim toga, iz literature je preuzeto dodatnih 80 primjera [Tai 03] za koje se broj poslova kreće od 15 do 100, a broj strojeva je 15 ili 20. Primjeri su definirani kao statička okolina u kojoj su svi poslovi pripravni od početka rada sustava.

Literatura

[Ada 02] T. P. Adams, *Creation of Simple, Deadline, and Priority Scheduling Algorithms using Genetic Programming,* Genetic Algorithms and Genetic Programming at Stanford 2002, http://www.genetic-programming.org/sp2002/Adams.pdf

[And 94] D. Andre, *Automatically Defined Features: The Simultaneous Evolution of 2-dimensional Featrue Detectors and an Algorithm for Using Them*, Advances in Genetic Programming, pp. 477-494, MIT Press, 1994.

[Atl 94] Bonnet L. Atlan, J.B. Polack, *Learning distributed reactive strategies by genetic programming for the general job shop problem,* Proceedings 7th annual Florida Artificial Intelligence Research Symposium, Pensacola, Florida, IEEE Press, 1994.

[Auy 03] Andy Auyeung, Iker Gondra, H. K. Dai, *Multi-heuristic list scheduling genetic algorithm for task scheduling,* Symposium on Applied Computing, Proceedings of the 2003 ACM symposium on Applied computing,Melbourne, Florida,Pages: 721 - 724, http://portal.acm.org/citation.cfm?doid=952532.952673

[Ban 02] W. Banzhaf, W. B. Langdon, *Some considerations on the reason for bloat,* Genetic Programming and Evolvable Machines, 3(1), pp. 81-91, March 2002.

[Ban 98] W. Banzhaf, P. Nordin, R. E. Kellet, F. D. Francone, *Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and its Applications*, Morgan Kaufmann, 1998.

[Bea 05] Open BEAGLE: a versatile EC framework, http://beagle.gel.ulaval.ca/

[Bea 90] J. E. Beasley, *OR-Library: Weighted tardiness*, 1990, http://people.brunel.ac.uk/~mastjjb/jeb/orlib/wtinfo.html

[Bli 94] T. Blickle, L. Thiele, *Genetic Programming and Redundancy,* Proc. Genetic Algorithms within the Framework of Evolutionary Computation (Workshop at KI-94), Saarbrücken, Germany, 1994. , http://www.handshake.de/user/blickle/publications/GPandRedundancy.ps

[Bli 96] Tobias Blickle, *Evolving Compact Solutions in Genetic Programming: A Case Study,* Parallel Problem Solving From Nature IV. Proceedings of the International Conference on Evolutionary Computation, LNCS, Vol. 1141, pp. 564-573, Springer-Verlag, 22-26 September 1996., http://www.blickle.handshake.de/publications/ppsn1.ps

[Bud 00] L. Budin, D. Jakobović, M. Golub, *Genetic Algorithms in Real-Time Imprecise Computing*, Journal of Computing and Information Technology CIT, Vol. 8, No. 3, September 2000, pp. 249-257.

[Bud 98] L. Budin, D. Jakobović, M. Golub, *Parallel Adaptive Genetic Algorithm*, Proc. Int. ICSC/IFAC Symposium on Neural Computation, NC'98, Vienna, September 23-25, 1998., pp. 157-163

[Bud 99] L. Budin, D. Jakobović, M. Golub, *Genetic Algorithms in Real-Time Imprecise Computing*, IEEE International Symposium on Industrial Electronics ISIE'99, Bled, 1999, Vol. 1, pp. 84-89.

[Bur 03] Edmund Burke, Steven Gustafson, Graham Kendall, *Ramped Half-n-Half Initialisation Bias in GP*, enetic and Evolutionary Computation -- GECCO-2003, LNCS, Vol. 2724, pp. 1800-1801, Springer-Verlag, 12-16 July 2003.

[Cha 04] Samarn Chantaravarapan, Jatinder N.D. Gupta, *Single Machine Group Scheduling with Setups to Minimize Total Tardiness,* 2004, http://www.pmcorp.com/PublishedPapers/Scheduling%20Publications/SingleMachineGroupSchedulingwithSetups.pdf

[Cha 96] Yih-Long Chang, Toshiyuki Sueyoshi, Robert Sullivan, *Ranking dispatching rules by data envelopment analysis in a job shop environment,* IIE Transactions, 28(8):631-642, 1996

[Che 99] V.H.L. Cheng, L.S. Crawford, P.K. Menon, *Air Traffic Control Using Genetic Search Techniques,* 1999 IEEE International Conference on Control Applications, August 22-27, Hawai'i, HA, http://www.optisyn.com/papers/1999/traffic_99.pdf

[Cic 01] Vincent A. Cicirello, Stephen F. Smith, *Ant Colony Control for Autonomous Decentralized Shop Floor Routing,* Fifth International Symposium on Autonomous Decentralized Systems March 26 - 28, 2001 Dallas, Texas p. 383, http://csdl.computer.org/comp/proceedings/isads/2001/1065/00/10650383abs.htm

[Cic 01a] Vincent Cicirello, Stephen Smith, *Randomizing Dispatch Scheduling Policies,* The 2001 AAAI Fall Symposium: Using Uncertainty Within Computation, November, 2001., http://www.ri.cmu.edu/pubs/pub_3789.html

[Cic 03] Vincent Cicirello, *Weighted Tardiness Scheduling with Sequence-Dependent Setups,* Technical report, The Robotics Institute, Carnegie Mellon Univeristy, 2003, http://www.cs.drexel.edu/~cicirello/benchmarks.html

[Cor 01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, 2nd ed., The MIT Press - McGraw-Hill, 2001.

[Cra 85] Nichael Lynn Cramer, *A Representation for the Adapative Generation of Simple Sequential Programs,* International Conference on Genetic Algorithms and their Applications [ICGA85], CMU, Pittsburgh, http://www.sover.net/~nichael/nlc-publications/icga85/index.html

[Dav 81] Ernest Davis, Jeffrey M. Jaffe, Algorithms for Scheduling Tasks on Unrelated Processors, Journal of the ACM, Volume 28 , Issue 4 (October 1981), pp. 721 – 736 http://portal.acm.org/citation.cfm?doid=322276.322284

[Dha 78] B. G. Dharan, T.E. Morton, Algoristics for Single Machine Sequencing with Precedence Constraints, Management Science 24, p. 1011-1020, 1978. http://www.ruf.rice.edu/~bala/files/dharan-morton-algoristics_for_sequencing-Mgt_Science_1978.pdf

[Dim 01] C. Dimopoulos, A. M. S. Zalzala, *Investigating the use of genetic programming for a classic one-machine scheduling problem,* Advances in Engineering Software, Volume 32, Issue 6 , June 2001, Pages 489-498, http://www.sciencedirect.com/

[Dim 99] Christos Dimopoulos, Ali M. S. Zalzala, *Evolving Scheduling Policies through a Genetic Programming Framework,* Proceedings of the Genetic and Evolutionary Computation Conference, Vol. 2, p. 1231, Morgan Kaufmann, 13-17 July 1999.

[Dim 99a] Dimopoulos C., Zalzala A.M.S., *A genetic programming heuristic for the one-machine total tardiness problem,* Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on , Volume: 3 , 6-9 July 1999

[Eib 00] Ágoston E. Eiben, Robert Hinterding, Zbigniew Michalewicz, *Parameter Control in Evolutionary Algorithms,* IEEE Trans. on Evolutionary Computation, 2000. http://citeseer.ist.psu.edu/eiben00parameter.html

[Gag 02] C. Gagné, M. Parizeau, *Open BEAGLE: A New Versatile C++ Framework for Evolutionary Computation*, Late Breaking papers at the Genetic and Evolutionary Computation Conference (GECCO-2002), New York, USA, 9-13 July 2002

[Gag 03] Christian Gagné, Marc Parizeau, Marc Dubreuil, *Distributed BEAGLE: An Environment for Parallel and Distributed Evolutionary Computations,* Proc. of the 17th Annual International Symposium on High Performance Computing Systems and Applications (HPCS) 2003, http://vision.gel.ulaval.ca/en/publications/Id_439/PublDetails.php

[Gat 97] C. Gathercole, P. Ross, *Tackling the Boolean Even N Parity Problem with Genetic Programming and Limited Error Fitness*, Genetic Programming 1997: Proceedings of the 2nd Annual Conference, pp. 119-127, San Francisco, 1997

[Gib 02] K. A. Gibbs, *Implementation and Evaluation of a Novel Branch Construct for Genetic Programming,* Genetic Algorithms and Genetic Programming at Stanford 2002, http://www.genetic-programming.org/sp2002/Gibbs.pdf

[Gol 00] M. Golub, D. Jakobović, *A New Model of Global Parallel Genetic Algorithm*, Proc. 22th Int. Conference ITI'00, Pula, June 13-16, 2000

[Gol 01] M. Golub, *Poboljšavanje djelotvornosti paralelnih genetskih algoritama*, doktorska disertacija, Fakulet elektrotehnike i računarstva, 2001.

[Gol 01a] M. Golub, D. Jakobović, L. Budin, *Parallelization of Elimination Tournament Selection without Synchronization*, Proc. 5th Int. Conf. on Intelligent Engineering Systems INES 2001, Helsinki, September 16-18., pp. 85-90., 2001.

[Gol 96] M. Golub, *Vrednovanje uporabe genetskih algoritama za aproksimaciju vremenskih nizova*, magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, 1996.

[Gol 98] M. Golub, D. Jakobović, *A Few Implementations Of Parallel Genetic Algorithm*, Proc. 20th Int. Conference ITI'98, Pula, June 14-17 1998, pp.332-337

[Gre 01] William A. Greene, *Dynamic Load-Balancing via a Genetic Algorithm,* 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01) November 07 - 09, 2001 Dallas, Texas, http://csdl.computer.org/comp/proceedings/ictai/2001/1417/00/14170121abs.htm

[Han 04] James V. Hansen, *Genetic search methods in air traffic control,* Computers and Operations Research, v 31, n 3, March, 2004, p 445-459, http://www.sciencedirect.com/

[Hay 96] T. D. Haynes, D. A. Schoenefeld, R. L. Wainwright, *Type Inheritance in Strongly Typed Genetic Programming*, Advances in Genetic Programming II, P. J. Angeline, K. E. Kinnear, (eds), MIT Press, 1996.

[He 03] Xiaoshan He, Xian-He Sun, Gregor Von Laszewski, *A QoS Guided Scheduling Model in Grid Environment*, Journal of Computer Science and Technology, Volume 18 , Issue 4 (July 2003), pp. 442 – 451 http://www.cs.iit.edu/~scs/psfiles/jcst_XHe-5-28.pdf

[Hel 02] Terry M. Helm, Steve W. Painter, Robert Oakes, *A comparison of three optimization methods for scheduling maintenance of high cost, long-lived capital assets,* Winter Simulation Conference Proceedings, v 2, 2002, p 1880-1884

[Hin 97] Robert Hinterding, Zbigniew Michalewicz, Agoston E. Eiben, *Adaptation in Evolutionary Computation: A Survey,* IEEECEP: Proceedings of The IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, 1997. http://citeseer.ist.psu.edu/hinterding97adaptation.html

[Iba 77] Oscar H. Ibarra, Chul E. Kim, *Heuristic Algorithms for Scheduling Independent Tasks on Nonidentical Processors*, Journal of the ACM, Volume 24 , Issue 2 (April 1977), pp. 280 – 289 http://portal.acm.org/citation.cfm?id=322011&jmp=abstract&dl=GUIDE&dl=ACM

[Jak 97] D. Jakobović, *Adaptive Genetic Operators in Elimination Genetic Algorithm*, Proc. 19th Int. Conference ITI'97, Pula, June 17-20 1997, pp.351-356

[Jak 98] D. Jakobović, M. Golub, *Adaptive Genetic Algorithm*, Proc. 20th Int. Conference ITI'98, Pula, June 14-17 1998, pp.351-356

[Jak 99] D. Jakobović, M. Golub, *Adaptive Genetic Algorithm*, Journal of Computing and Information Technology CIT, Vol. 7, No. 3, September 1999., pp. 229-236

[Jon 98] Albert Jones, Luis C. Rabelo, *Survey of Job Shop Scheduling Techniques,* NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, 1998., http://www.nist.gov/msidlibrary/summary/9820.html

[Kas 99] Joachim Käschel, Tobias Teich, Gunnar Köbernik, Bernd Meier, *Algorithms for the Job Shop Scheduling Problem - a comparison of different methods*, European Symposium on Intelligent Techniques ESIT ´99, June 3-4, 1999, Orthodox Academy of Crete, Greece http://www.erudit.de/erudit/events/esit99/12553_P.pdf

[Kin 93] Kenneth E. Kinnear, *Evolving a Sort: Lessons in Genetic Programming,* Proceedings of the 1993 International Conference on Neural Networks, Vol. 2, pp. 881-888, IEEE Press, 28 March -1 April 1993., ftp://cs.ucl.ac.uk/genetic/ftp.io.com/papers/kinnear.icnn93.ps.Z

[Koz 03] J. R. Koza, M. A. Keane, M. J. Streeter, W. Mydlowec, J. Yu, G. Lanza, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*, Kluwer Academic Publishers, 2003.

[Koz 90] John R. Koza, *Genetically Breeding Populations of Computer Programs to Solve Problems in Artificial Intelligence ,* Proceedings of the Second International Conference on Tools for AI, Herndon,

Virginia, USA, 1990
http://citeseer.ist.psu.edu/koza90genetically.html

[Koz 90a] John R. Koza, *Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems,* Stanford University Computer Science Department technical report STAN-CS-90-1314. June 1990.,
http://www.genetic-programming.com/jkpubs72to93.html

[Koz 92] J. R. Koza, *Genetic Programming – On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[Koz 94] J. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*, MIT Press, 1994.

[Koz 95] J. Koza, *Genetic Programming and Hill Climbing*, Machine Learning List, Vol. 7, No. 14, 18.9.1995.
http://www.ics.uci.edu/~mlearn/MLlist/v7/14.html

[Lan 00] W. B. Langdon, W. Banzhaf, *Genetic Programming Bloat without Semantics,* Parallel Problem Solving from Nature - PPSN VI 6th International Conference, LNCS, Vol. 1917, pp. 201-210, Springer Verlag, 16-20 September 2000.,
ftp://cs.ucl.ac.uk/genetic/papers/wbl_ppsn2000.ps.gz

[Lan 02] W. B. Langdon, R. Poli, *Foundations of Genetic Programming*, Springer-Verlag, 2002.

[Lan 05] William Langdon, Steven Gustafson, John Koza, *The Genetic Programming Bibliography*, 2005
http://liinwww.ira.uka.de/bibliography/Ai/genetic.programming.html

[Lan 97] W. B. Langdon, R. Poli, *Genetic Programming Bloat with Dynamic Fitness,* Technical Report, University of Birmingham, School of Computer Science, Number CSRP-97-29, 3 December 1997.,
ftp://ftp.cs.bham.ac.uk/pub/tech-reports/1997/CSRP-97-29.ps.gz

[Lan 98] W. B. Langdon, *Genetic Programming and Data Structures*, Kluwer Academic Publishers, 1998.

[Lee 04] S. M. Lee, A.A. Asllani, *Job scheduling with dual criteria and sequence-dependent setups: mathematical versus genetic programming,* Omega, v 32, n 2, April 2004, p 145-53

[Lee 97] Young Hoon Lee, Kumar Bhaskaran, Michael Pinedo, *A heuristic to minimize the total weighted tardiness with sequence-dependent setups,* IIE Transactions, 29, 45-52, 1997.

[Lek 03] Lekin®, Flexible Job Shop Scheduling System
http://www.stern.nyu.edu/om/software/lekin/

[Leu 04] J. Y-T. Leung (ed.), *Handbook of scheduling,* Chapman & Hall/CRC, 2004.

[Leu 95] J. Y-T. Leung, *A survey of scheduling results for imprecise computation tasks*, Imprecise and approximate computation, Kluwer Academic Publishers, pp. 35-42, 1995

[Lop 01] A. Lopez-Ortiz, *Computational Theory FAQ*,
http://db.uwaterloo.ca/~alopez-o/comp-faq/faq.html

[Mar 04] Goran Martinović, *Postupci raspoređivanja u raznorodnim računalnim sustavima*, doktorska disertacija, Fakultet elektrotehnike i računarstva, Zagreb, 2004.

[Meg 05] Nicole Megow, Marc Uetz, Tjark Vredeveld, *Stochastic Online Scheduling on Parallel Machines,* G. Persiano and R. Solis-Oba (eds): Approximation and Online Algorithms, Lecture Notes in Computer Science 3351, pages 167-180, Springer, 2005.,
http://www.math.tu-berlin.de/~nmegow/muv05sos.pdf

[Mic 92] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programs*, Springer-Verlag, Berlin, 1992

[Miy 00] Kazuo Miyashita, *Job-Shop Scheduling with GP,* Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000), pp. 505-512, Morgan Kaufmann, 10-12 July 2000.

[Moh 83] Ram Mohan, V. Rachamadugu, Thomas E. Morton, *Myopic Heuristics for the Weighted Tardiness Problem on Identical Parallel Machines,* Working Paper, The Robotics Institute, Carnegie-Mellon University, 1983.

[Mon 01] Patrick Monsieurs, Eddy Flerackers, *Detecting and Removing Inactive Code in Genetic Programs,* 2001,
http://alpha.luc.ac.be/~lucp1089/DetectingAndRemovingInactiveCode.pdf

[Mon 95] D. J. Montana, *Strongly Typed Genetic Programming*, Evolutionary Computation, 3(2):199-230, 1995.

[Mor 93] Thomas E. Morton, David W. Pentico, *Heuristic Scheduling Systems,* John Wiley & Sons, Inc., 1993.

[Nei 99] Michael O'Neill, Conor Ryan, *Automatic Generation of Caching Algorithms,* Evolutionary Algorithms in Engineering and Computer Science, pp. 127-134, John Wiley & Sons, 30 May - 3 June 1999., http://www.mit.jyu.fi/eurogen99/papers/oneill.ps

[Nei 99a] Michael O'Neill, Conor Ryan, *Automatic Generation of Programs with Grammatical Evolution*, 1999, http://citeseer.ist.psu.edu/276159.html

[Nor 94] P. Nordin, *A Compiling Genetic Programming System that Directly Manipulates the Machine Code*, Advances in Genetic Programming, pp. 311-331, MIT Press, 1994

[Nor 95] P. Nordin, W. Banzhaf, *Genetic Programming Controlling a Miniature Robot*, Working Notes for the AAAI Symposium on Genetic Programming, pp. 61-67, MIT, Cambridge, 1995.

[Nov 03] Sonja Novković, Davor Šverko, *A Genetic Algorithm With Self-Generated Random Parameters,* Journal of Computing and Information Technology - CIT, Vol. 11, No. 4, December 2003., pp. 271-284

[Ok 00] S. Ok, K. Miyashita, S. Nishihara, *Improving Performance of GP by Adaptive Terminal Selection,* Proc. of the Pacific Rim International Conference on Artificial Intelligence (PRICAI), pp.435-445, 2000, http://staff.aist.go.jp/k.miyashita/publications/PRICAI2000.ps

[Ok 01] S. Ok, K. Miyashita, K. Hase, *Evolving Bipedal Locomotion with Genetic Programming --- Preliminary Report,* Proc. of the Congress on Evolutionary Computation 2001, pp.1025-1032, 2001, http://staff.aist.go.jp/k.miyashita/publications/cec.ps

[Pat 97] Norman Paterson, Mike Livesey, *Evolving caching algorithms in C by genetic programming,* Genetic Programming 1997: Proceedings of the Second Annual Conference, pp. 262-267, Morgan Kaufmann, 13-16 July 1997., http://www.dcs.st-and.ac.uk/~norman/Pubs/cache.ps.gz

[Pen 00] Carlos Andrés Peña-Reyes, Moshe Sipper, *Evolutionary computation in medicine: an overview,* Artificial Intelligence in Medicine Volume 19, Issue 1, 1 May 2000, Pages 1-23, http://www.sciencedirect.com/

[Pin 04] M. Pinedo, *Offline Deterministic Scheduling, Stochastic Scheduling, and Online Deterministic Scheduling: A Comparative Overview*, Handbook of Scheduling, J. Y-T. Leung (ed.), Chapman & Hall/CRC, 2004.

[Pol 99] Riccardo Poli, *Parallel Distributed Genetic Programming,* New Ideas in Optimization, McGraw-Hill, 1999., http://citeseer.ist.psu.edu/328504.html

[Pru 04] K. Pruhs, J. Sgall, E. Torng, *Online scheduling*, Handbook of Scheduling, J. Y-T. Leung (ed.), Chapman & Hall/CRC, 2004.

[Rus 97] R. M. Russell, J. E. Holsenback, *Evaluation of greedy, myopic and less-greedy heuristics for the single machine, total tardiness problem,* Journal of the Operational Research Society (1997), 48, 640-646

[Sch 94] E. Schöneburg, F. Heinzmann, S. Feddersen, *Genetische Algorithmen und Evolutionsstrategien*, Addison-Wesley, 1994.

[Ser 01] F. Seredynski, J. Koronacki, C. Z. Janikow, *Distributed multiprocessor scheduling with decomposed optimization criterion,* Future Generation Computer Systems, Volume 17, Issue 4 , January 2001, Pages 387-396, http://www.sciencedirect.com/

[Ser 99] F. Seredynski, J. Koronacki, C. Z. Janikow, *Distributed Scheduling with Decomposed Optimization Criterion: Genetic Programming Approach,* Internationa Parallel and Distributed Processing Symposium, workshop: Bio-Inspired Solutions to Parallel Processing Problems, 1999., http://ipdps.eece.unm.edu/1999/biosp3/seredyns.pdf

[Sil 03] Sara Silva, Jonas Almeida, *Dynamic Maximum Tree Depth - A Simple Technique for Avoiding Bloat in Tree-Based GP ,* Proc. of the Genetic and Evolutionary Computation Conference (GECCO-2003), pp. 1776-1787, Genetic and Evolutionary Computation Conference (GECCO-2003), Chicago, Illinois USA,

July-2003,
http://cisuc.dei.uc.pt/ecos/view_pub.php?id_p=109

[Sou 98] T. Soule, *Code Growth in Genetic Programming,* PhD Thesis, University of Idaho, 1998., http://www.cs.uidaho.edu/~tsoule/research/the3.ps

[Sri 94] M. Srinivas, L. M. Patnaik, *Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms*, IEEE Trans. Systems, Man and Cybernetics, April 1994.

[Sri 94a] M. Srinivas, L. M. Patnaik, *Genetic Algorithms: A Survey*, IEEE Computer, June 1994.

[Tac 94] W. A. Tackett, *Recombination, Selection and the Genetic Construction of Computer Programs*, PhD thesis, University of Southern California, Department of Electrical Engineering Systems, 1994.

[Tai 03] E. Taillard, *Scheduling Instances*, 2003. http://ina.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html

[Tal 03] W. A. Talbott, *Automatic Creation of Team-Control Plans Using an Assignment Branch in Genetic Programming,* Genetic Algorithms and Genetic Programming at Stanford 2003, http://www.genetic-programming.org/sp2003/Talbott.pdf

[Tel 95] A. Teller, M. Veloso, *PADO: Learning Tree Structured Algorithms for Orchestration into an Object Recognition System*, Technical Report CMU-CS-95-101, Department of Computer Science, Carnegie Mellon University, 1995.

[Vaz 00] Manuel Vazquez, L. Darrell Whitley, *A Comparison of Genetic Algorithms for the Dynamic Job Shop Scheduling Problem,* Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '00), Las Vegas, Nevada, USA, July 8-12, 2000

[Wal 05] Scott S. Walker, Robert W. Brennan, Douglas H. Norrie, *Holonic Job Shop Scheduling Using a Multiagent System*, IEEE Intelligent Systems, 2/2005, pp. 50-57

[Wal 96] P. Walsh, C. Ryan, *Paragen: A Novel Technique for the Autoparallelisation of Sequential Programs Using Genetic Programming*, Genetic Programming 96: Proceedings of the 1st Annual Conference, pp. 406-409, MIT Press, 1996.

[Wan 03] J.-S. Wang, *Influences of Function Sets in Genetic Programming,* Genetic Algorithms and Genetic Programming at Stanford 2003, http://www.genetic-programming.org/sp2003/Wang.pdf

[Wol 97] D. H. Wolpert, W. G. Macready, *No Free Lunch Theorems for optimization*, IEEE Trans. on Evolutionary Computation, 1(1):67-82, 1997.

[Yin 03] Wen-Jun Yin, Min Liu, Cheng Wu, *Learning single-machine scheduling heuristics subject to machine breakdowns with genetic programming,* Proceedings of the 2003 Congress on Evolutionary Computation CEC2003, pp. 1050-1055, IEEE Press, 8-12 December 2003.,

[Zha 96] B.-T. Zhang, H. Mühlenbein, *Adaptive Fitness Functions for Dynamic Growing/Pruning, of Program Trees*, Advances in Genetic Programming 2, pogl. 12, pp.241-256, MIT Press, 1996.