Genetic programming: An introduction and survey of applications

Conference Paper · October 1997

DOI: 10.1049/cp:19971199 · Source: IEEE Xplore

5 authors, including:

Mark J Willis
Newcastle University

Hugo Hiden
Newcastle University

Ben McKay
Avantium

Gary Montague
Teesside University

GENETIC PROGRAMMING: AN INTRODUCTION AND SURVEY OF APPLICATIONS

M.J. Willis*, H.G Hiden*, P. Marenbach+, B. McKay* and G.A. Montague*

* Symbolic Optimisation Research Group (SORG)
Dept. of Chemical and Process Engineering
University of Newcastle upon Tyne
NE1 7RU, UK

\+ Institute of Control Engineering
Darmstadt University of Technology
Landgraf-Georg-Strasse 4
D-64283 Darmstadt, Germany

{Mark.Willis, H.G.Hiden, Ben.McKay, Gary.Montague}
@ncl.ac.uk
http://lorien.ncl.ac.uk/sorg

mali@rt.e-technik.th-darmstadt.de

Keywords: genetic programming, survey

Abstract

The aim of this paper is to provide an introduction to the rapidly developing field of genetic programming (GP). Particular emphasis is placed on the application of GP to engineering problem solving. First, the basic methodology is introduced. This is followed by a review of applications in the areas of systems modelling, control, optimisation and scheduling, design and signal processing. The paper concludes by suggesting potential avenues of research.

Introduction

GP began as an attempt to discover how computers could learn to solve problems without being explicitly programmed to do so. The GP technique is an evolutionary algorithm that bears a strong resemblance to genetic algorithm's (GA's). The primary differences between GA's and GP can be summarised as follows;

- GP typically codes solutions as tree structured, variable length chromosomes, while GA's generally make use of chromosomes of fixed length and structure.
- GP typically incorporates a domain specific syntax that governs acceptable (or meaningful) arrangements of information on the chromosome. For GA's, the chromosomes are typically syntax free.
- GP makes use of genetic operators that preserve the syntax of its tree-structured chromosomes during 'reproduction'.
- GP solutions are often coded in a manner that allows the chromosomes to be executed directly using an appropriate interpreter. GA's are rarely coded in a directly executable form.

The use of this flexible coding system allows the algorithm to perform structural optimisation. This can be useful for the solution of many engineering problems. For instance, GP may be used to perform symbolic regression. While conventional regression seeks to optimise the parameters for a pre-specified model structure, with symbolic regression, the model structure and parameters are determined simultaneously. Similarly, the evolution of control algorithms, scheduling programs, structural design and signal processing algorithms can be viewed as structural optimisation problems suitable for GP.

Cramer (1985) developed one of the first tree structured GA's for basic symbolic regression. Another early development was the BEAGLE[1] algorithm of Forsyth, (1986), which generated classification rules using a tree structured GA. However, it was Koza (1992 and 1994) who was largely responsible for the popularisation of GP within the field of computer science. His GP algorithm (coded in LISP) was applied to a wide range of problems including symbolic regression, control, robotics, games and classification.

Since this initial work, interest in the field has grown, with the first international conference on GP held at Stanford University in 1996 (GP'96). While still dominated by computer scientists, engineering applications have begun to appear. Therefore, the objective of this paper is to discuss these recent engineering applications and provide an entry point to this rapidly expanding field.

The paper is organised as follows. First, GP is introduced. Next, a survey of engineering applications within the GP field is provided. Finally, the paper concludes with the authors' perspective on future research directions.

[1]BEAGLE - Biological Evolutionary Algorithm Generating Logical Expressions

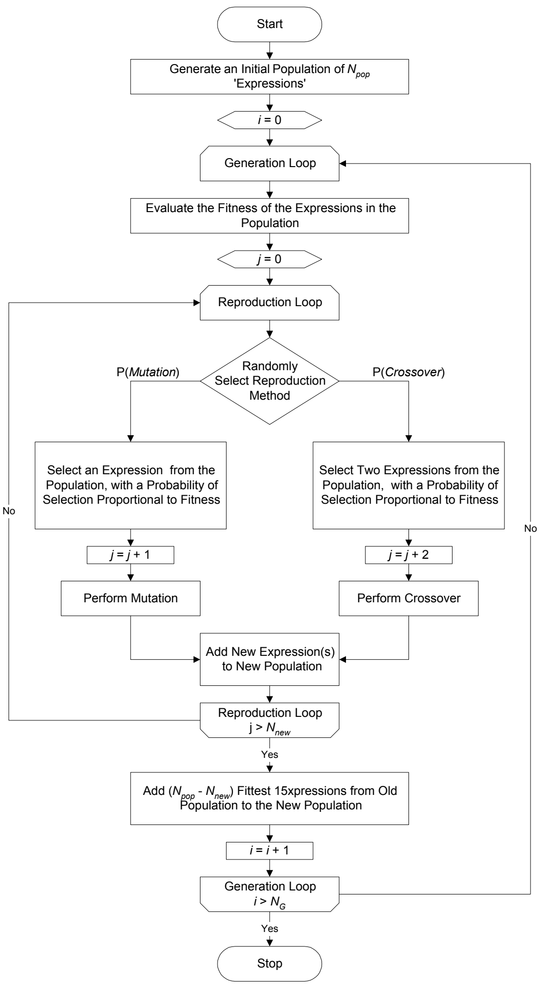

Figure 1: Typical genetic programming algorithm flowsheet.

Genetic Programming

A GP algorithm works on a population of individuals, each of which represent a potential solution to a problem. A flowchart of a typical GP algorithm is shown in Fig. 1. In order to solve a problem using GP Koza (1992) states that it is necessary to specify the following;

- **The terminal set**: A set of input variables or constants.

- **The function set:** A set of domain specific functions used in conjunction with the terminal set to construct potential solutions to a given problem. For symbolic regression this could consist of a set of basic mathematical functions, while Boolean and conditional operators could be included for classification problems.

- **The fitness function:** Fitness is a numeric value assigned to each member of a population to provide a measure of the appropriateness of a solution to the problem in question.

- **The algorithm control parameters:** This includes the population size and the crossover and mutation probabilities.

- **The termination criterion:** This is generally a pre-defined number of generations or an error tolerance on the fitness.

It should be noted that the first 3 components determine the algorithm search space, while the final 2 components affect the quality and speed of search.

In order to further illustrate the coding procedure and the genetic operators used for GP, a symbolic regression example will be used. Consider the problem of predicting the numeric value of an output variable, *y*, from two input variables *a* and *b*. One possible symbolic representation for *y* in terms of *a* and *b* would be,

$$y = (a - b) / 3 \tag{1}$$

Figure (2) demonstrates how this expression may be represented as a tree structure[2].

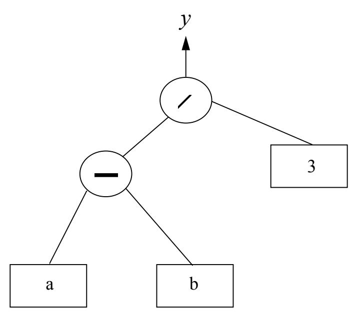

Figure 2: Representation of a numeric expression using a tree structure.

With this tree representation, the genetic operators of crossover and mutation must be posed in a fashion that allows the syntax of resulting expressions to be preserved. Figure (3) shows a valid crossover operation where the two parent expressions are given by:

Parent 1:	$y = (a - b) / 3$	(2)
Parent 2:	$y = (c - b) * (a + c)$	(3)

Parent 1 has input variables '*a*' and '*b*' and a constant '*3*' while parent 2 has three input variables '*a*', '*b*' and '*c*'. Both expressions attempt to predict the process output, '*y*'. If the '/' from parent 1 and the '*' from parent 2 are chosen as the crossover points, then the two offspring are given by:

Offspring 1:	$y = (a - b) / (a + c)$	(4)
Offspring 2:	$y = (c - b)* 3$	(5)

[2] For a thorough discussion of tree structures and their properties, see Tenenbaum and Augenstein (1981).

It is assumed that by recombining relevant sub-trees, it is possible to produce new expressions that provide fitter solutions.
In order to provide population diversity and allow the exploration of areas of the solution space not represented in the initial population, a mutation operator may also be used. Mutation merely consists of randomly changing a function, input or constant in one of the mathematical expressions making up the present population.

GP Applications

The following section presents a review of engineering applications of GP. The results of the literature survey have been arranged into the following broad categories:

- Systems Modelling
- Control
- Optimisation and scheduling
- Design
- Signal processing

<u>Systems Modelling</u>

In Koza (1992), basic regression examples were used to illustrate the functionality of a GP algorithm. The examples included the discovery of trigonometric identities, polynomial approximation and basic econometric modelling and forecasting. Iba *et al.* (1993 and 1994) introduced a GP algorithm, called STROGANOFF[3]. The algorithm integrated multiple regression analysis and a GA-based search strategy. The function set was limited to quadratic polynomials in two variables. The effectiveness of STROGANOFF was demonstrated by solving several system identification problems. Oakley (1994) and Howard and Oakley (1995) used Koza's LISP code (Koza, 1992) to demonstrate that GP can be used to perform time series prediction of chaotic systems.

Applications to chemical process systems have included the generation of non-linear dynamic models of biotechnological batch and fed-batch fermentations (Bettenhausen and Marenbach, 1995; Bettenhausen *et al.*., 1995b; Marenbach *et al.*, 1996), the identification of complex fluid flow patterns (Watson and Parmee, 1996) and the generation of steady-state input-output models of a range of industrial chemical process systems (McKay *et al.*, 1995, 1996b and 1997b). Comparisons with established modelling paradigms (such as neural networks) are also available (McKay et al., 1996a).

Bettenhausen *et al.* (1995b) and Marenbach *et al.* (1996) developed a GP algorithm (referred to as a structured model generator or SMOG[4]) that models dynamic processes by allowing state variables and feed-back loops. Such a system appears to have very powerful representational capabilities. Gray et al. (1996a) also successfully use a block diagram oriented approach for non-linear system identification.

A number of contributions have reported a successful 'hybrid' of mechanistic approaches and GP techniques for process modelling. For instance the work of Gray *et al*. (1996b) on the modelling of a simple simulated water tank example. In more recent work, Elsey *et al*. (1997), apply a hybrid GP approach for the development of a cooking extrusion process (using GP to develop rheological models for an otherwise mechanistic extruder model), while McKay *et al.*(1997a) uses a hybrid GP technique for the modelling of a fed-batch fermentation process.

<u>Control</u>

Early applications in the area of control include the evolution of non-linear control strategies for broom balancing (Koza and Keane, 1990). Hampo (1992) and Hampo and Marko (1992) considered the use of GP for developing vehicle control systems, with emphasis placed on active suspension control. A popular subject in the GP literature is the development of robot control strategies (e.g. see Handly, 1993 and 1994; Ghanea-Herrock and Fraser ,1994; Gruau and Quatramaran ,1996; and Nordin and Banzhaf ,1996). Alba *et al.*(1996) use GP to develop rule bases for the definition of fuzzy logic controllers. Steinkolger and Koch (1996) also report the use of genetic programming techniques for the development of hierarchical fuzzy logic controllers.

<u>Optimisation and Scheduling</u>

Grimes (1995), used GP to plan the maintenance schedule for a length of railway track. GP was used to generate rules determining when track maintenance was required on a particular section of track. GP was compared with existing maintenance scheduling tools and found to provide superior performance. Montana and Czerwinski (1996) employed a GP algorithm to optimally control the timing of traffic signals within a network of interconnecting road junctions. The timing sequences generated varied with the level of congestion and the waiting time for cars at various junctions. The systems developed using GP were tested using a number of different road configurations under different levels of congestion, and found to outperform schemes based on fixed traffic signal timings. Gaces-Perez *et al.*(1996), implemented a GP based facility layout scheme, whereby

---

[3] STructured Representation On Genetic Algorithms for Non-linear Function Fitting

[4] SMOG evolves ‘signal path models’ (similar to the block diagrams used by control engineers) from a function set of mathematical functions and linear transfer function blocks. Recurrent connections allow the inclusion of recycle loops.

a number of rectangular facilities were optimally placed within a fixed site boundary.

Design

The application of GP to the design of new polymeric materials was addressed by Porter *et al.*(1996). Here, the GP performed a structural optimisation of a monomer in order to achieve desired polymer properties. Koza *et al.*(1996a) described an automated process for designing electrical circuits. GP produced both the topology of the desired circuit and the sizing for all of the components in the circuit. Koza *et al.*(1996b) demonstrated the usefulness of automatically defined functions (ADF's) and architecture altering operations for designing analog electrical circuits.

Signal Processing

GP has also been used by Sharman *et al.*(1995) and Sharman and Esparcia-Alcazar (1996) to evolve the structure and parameters of adaptive digital signal processing (DSP) algorithms. In this application, GP was used to evolve the structure of the algorithm, while the parameters contained within it were optimised using simulated annealing. The GP based approach was compared with a more traditional recursive least squares based adaptive algorithm. It was concluded that the method is a versatile tool, applicable to a wide variety of signal processing applications.

Conclusions

This survey paper has revealed that engineering applications of GP are currently focused on typical systems problems such as modelling, control and optimisation. While computer scientists have concentrated on gaining a fundamental understanding of the algorithm (and improving its performance) the engineering community is addressing practical issues, often by introducing accepted systems engineering concepts and methodologies. For instance, the incorporation of local hill climbing for parameter optimisation and the use of cross validation techniques to ensure model generalisation.

Perhaps the most promising research direction appears to be the application of GP techniques to engineering design problems. While computational considerations currently limit the complexity of design applications that can be addressed, these will inevitably be lifted as processor speeds continue to increase.

Further potential avenues of research include the investigation of other algorithms capable of performing structural optimisation. For instance, the structural annealing algorithm of O'Reilly and Oppacher (1994), which is similar to GP but uses a population of one and a simulated annealing style mutation operator. This is reported to achieve similar performance to GP and as such warrants further investigation.

It is emphasised that GP is a young field of research, whose practitioners are still exploring its capabilities and limitations. Consequently it is the authors' belief that the future holds much promise.

References

Alba, E., Cotta, C. and Troyo, J.J., (1996), 'Type constrained genetic programming for rule based definition in fuzzy logic controllers', *Proc. First Anual. Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.255 - 260.

Bettenhausen, K.D., Marenbach, P., Freyer, S., Rettenmaier, H. and Nieken, U., (1995b), 'Self-organising structured modelling of a biotechnological fed-batch fermentation by means of genetic programming', *Proc. IEE Conf. on Genetic Algorithms in Engng. Systems: Inovations and Applications - GALESIA'95*, Sheffield, UK, No 414, pp.481-486.

Bettenhausen, K.D. and Marenbach, P., (1995), 'Self-organising modelling of biotechnological batch and fed-batch fermentations', *Proc. EUROSIM'95*.

Cramer, N.L., (1985), 'A representation for the adaptive generation of simple sequential programs'. *Proc. Int. Conf. on Genetic Algorithms and their Applications*, Carnegia Mellon University, Pittsburgh, USA, pp.183-187.

Elsey, J., Riepenhausen, J., McKay, B., Barton G.W. and Willis M.J., (1997), 'Modelling and control of a food extrusion process', *To be presented at Joint Int. Symp. on Process Systems Engng. and European Symp. on Comp. Aided Design - PSE'97/ESCAPE 7*, Trondheim, Norway.

Forsyth, R., (1986), 'Evolutionary learning strategies', Forsyth, R. and Rada, R. (Ed.), *Machine Learning: Applications in Expert Systems and Information Retrieval*, Ellis Horwood Ltd., USA, pp.79-95.

Garces-Perez, J., Schoenefeld, D.A. and Wainwright, R.L. (1996), 'Solving facility layout problems using genetic programming', *Proc. of the First Annual Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.182-190.

Ghanea-Herrock, R. and Fraser, A.P., (1994), 'Evolution of autonomous robot control architectures', *Evolutionary Computation*.

Gray, G.J., Murray-Smith, D.J., Li, Y. and Sharman, K.C. (1996a), 'Structural system identification using genetic programming and a block diagram oriented simulation tool'. *Electronic Letters*, Vol.32, No.15, pp.1422-1424.

Gray, J., Murry-Smith, D.J., Yun, L. and Sharman, K.C., (1996b), 'Nonlinear model structure identification using genetic programming', *Late Breaking Papers at the Genetic Programming 1996 Conference*, Koza, J.R (ed.), Stanford University Bookstore, USA, pp.32-37.

Grimes, C.A., (1995), 'Application of genetic techniques to the planning of railway track maintenance' *Proc. IEE Conf. on Genetic Algorithms in Engng. Systems: Inovations and Applications - GALESIA'95*, Sheffield, UK, pp.467-472.

Gruau, F. and Quatramaran, K., (1996), 'Cellular encoding for interactive evolutionary robotics', *Cognitive science research paper*, No.425.

Hampo, R.J. (1992) 'Genetic programming: A new paradigm for control and analysis', *Proc. 3rd ASME Symposium on Transportation Systems*, Anaheim, USA, pp.155-163.

Hampo, R.J. and Marko, K.A., (1992) 'Application of Genetic Programming to the control of vehicle systems', Proc. Of the intelligent vehicles symposium, June 29-July 1, Detroit, Mi, USA

Handly, S., (1993), 'The genetic planner: The automatic generation of plans for a mobile robot via genetic programming', *Proc. 8th IEEE Int. Symp. on Intelligent Control*, Chicago, USA.

Handly, S., (1994), 'The automatic generation of plans for a mobile robot via genetic programming with automatically defined functions', Kinnear, K.E. (Ed.), *Advances in Genetic Programming*, 18, pp.391-407.

Howard, E. and Oakley, N., (1995), 'Genetic programming as a means of assessing and reflecting chaos', *Tech. Report FS-95-01*, AAAI Press, pp.68-72.

Iba, H., Kurita,T., Garis,H. and Sato,T., (1993), 'System identification using structured genetic algorithms', *Proc. 5th Int. Conf. on Genetic Algorithms*, Urbana-Champaign, USA, pp.276-286.

Iba, H., de Garis, H. and Sato, T., (1994), 'Genetic programming using local hill-climbing', Davidor, Y. ,Schwefel, H.P. and Manner, R. (Ed.), *Parallel Problem Solving in Nature - PPSN III*, Vol.866 of Lecture Notes in Computer Science, Springer-Verlag, Germany, pp.302-311.

Koza, J. R. and Keane, M.A., (1990), 'Genetic breeding of non-linear optimal control strategies for broom balancing', *Proc. 9th Int. Conf. on Analysis and Optimisation of Systems*, France, pp.47-56.

Koza, J., (1992), 'Genetic programming: On the programming of computers by means of natural selection', The MIT Press, USA.

Koza, J., (1994), *Genetic Programming II: Automatic Discovery of Reusable Programs*, The MIT Press, USA.

Koza J.R., Bennett, F.H., Andre, D. and Keane, M.A., (1996a) 'Automated WYWIWYG design of both the topology and component values of electrical circuits using genetic programming', *Proc. of the First Annual Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.123-131.

Koza J.R., Andre, D., Bennett, F.H. and Keane, M.A., (1996b), 'Use of automatically defined functions and architecture altering operations in automated circuit synthesis with genetic programming', *Proc. of the First Annual Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.132-140.

Marenbach, P., Bettenhausen, K.D. and Freyer, S., (1996), 'Signal path oriented approach for generation of dynamic process models', *Proc. 1st Anual Conf. on Genetic Programming*, Stanford University, USA, pp.327-332.

McKay,B., Willis, M.J. and Barton, G. (1995). 'On the application of genetic programming to chemical process systems', *Proc. IEEE Int. Conf. on Evolutionary Computation* - ICEF'95, Perth, Australia.

McKay, B., Lennox, B., Willis, M.J., .Montague, G. and Barton, G.W., (1996a), 'Extruder modelling: A comparison of two paradigms', *Proc. of the UKACC Int. Conf. on Control - CONTROL-96*, Exeter, UK, Vol.2, pp.734-739.

McKay,B., Elsey,J., Willis,M.J. and Barton,G.W., (1996b), 'Evolving input-output models of chemical process systems using genetic programming', *Proc. IFAC, 13th World Congress* '96, San Francisco, USA, Vol.M, pp.277-282.

B. McKay, C. Sanderson, M.J. Willis, J. Barford, G. Barton., (1997a), 'Evolving a hybrid model of a fed-batch fermentation process', *Accepted for publication in Trans. of the Inst. of Measurement and Control*.

McKay,B., Willis, M.J., Barton, G., (1997b), 'Steady_state modelling of chemical process systems using genetic programming', *Accepted for publication in Comp. and Chem. Eng*.

Montana, D.J., Czerwinski, S., (1996), 'Evolving control laws for a network of traffic signals', *Proc. First Anual Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.333-338.

Nordin, P. and Banzhaf, W., (1996), 'Genetic programming controlling a miniature robot', *Proc. First Anual Conf. on Genetic Programming - GP'96*, Stanford University, USA, pp.61 - 67.

Oakley, N., (1994) 'Two scientific applications of genetic programming: stack filter and non-linear equation fitting to chaotic data', Kinnear, K.E. (Ed.), *Advances in Genetic Programming*, Cambridge, Mass: MIT Press, pp 369-389.

O'Reilly, U.M. and Oppacher, F., (1994), 'Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing', Davidor, Y. ,Schwefel, H.P. and Manner, R. (Ed.), *Parallel Problem Solving in Nature - PPSN III*, Vol.866 of Lecture Notes in Computer Science, Springer-Verlag, Germany, pp.397-406.

Porter, M. Willis, M. and Hiden, H.G. (1996), 'Computer-Aided Polymer Design Using Genetic Programming', *MEng. Research Project, Dept. Chemical and Process Engng, Uni. of Newcastle,* UK.

Tenenbaum, A.M. and Augenstein, M.J., (1981), *Data structures using pascal*, Prentice Hall Inc, USA.

Steinkolger, A. and Koch, J., (1996), ‘Genetic programming designs hierarchic fuzzy logic controllers’, *Proc. Fuzzy Logic in Engineering and Natural Sciences – FUZZY’96*, Zittav, Germany.

Sharman, K.C., Esparcia-Alcazar, A.I., Li, Y., (1995), 'Evolving signal processing algorithms by genetic programming'. *Proc. IEE Conf. on Genetic Algorithms in Engng. Systems: Inovations and Applications - GALESIA’95*, Sheffield, UK, No 414, pp.473-480.

Esparcia-Alcazar, A.I. and Sharman, K.C., (1996), 'Some applications of genetic programming in digital signal processing', *Late Breaking Papers at the Genetic Programming 1996 Conference*, Koza, J.R (Ed.), Stanford University Bookstore, USA, pp.24-31.

Watson, A.H., Parmee, I.C., (1996), ‘Identification of fluid systems using genetic programming’, *Proc. of the Second Online Workshop on Evolutionary Computation - WEC2*, pp.45-48

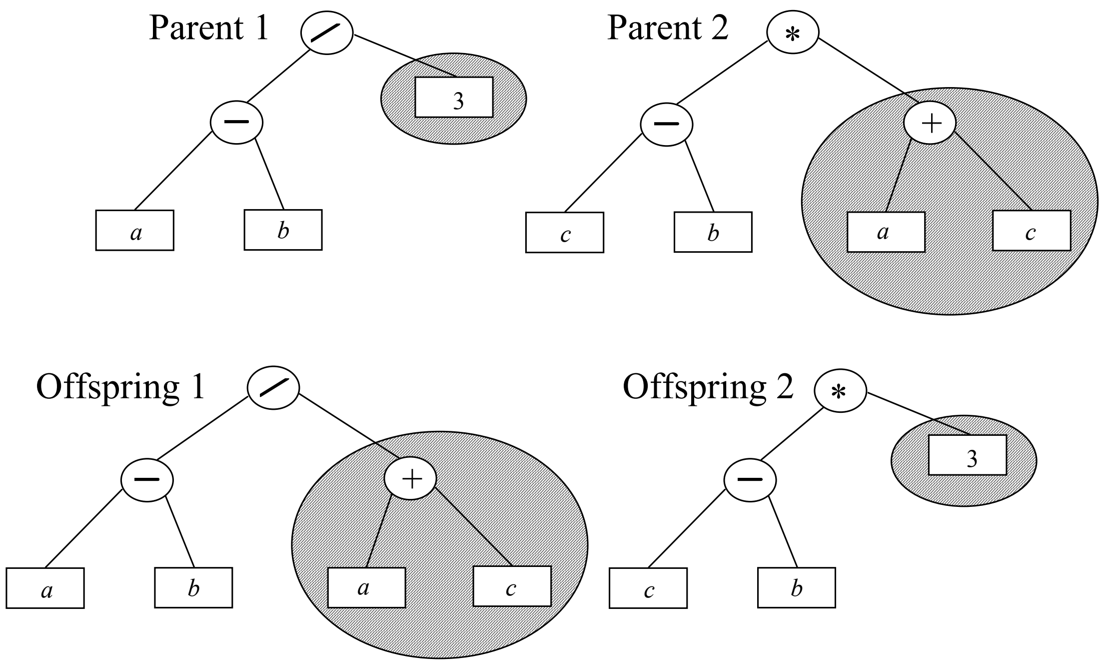

Figure 3. A typical crossover operation.