Verteilte Algorithmen: Nutzen sie der Forschergruppe?

Wolfgang Reisig

Humboldt-Universität zu Berlin
Institut für Informatik
Unter den Linden 6
D-10099 Berlin

Der Begriff „Verteilte Algorithmen“ bezeichnet Algorithmen kleinen und mittleren Umfangs, die durch konzentriertes Hinschauen oder durch händische systematische Konstruktion verständlich und nachvollziehbar sind. Zahlreiche solcher schwierigen Algorithmen enthalten die in der Literaturliste genannten Bücher.

Beispiele für kleine und mittelgroße Algorithmen sind der Echo-Algorithmus und — an der Grenze des derzeit Beherrschbaren — der Algorithmus von Gallagher, Humblet und Spira (GHS-Algorithmus). Auch wenn der GHS-Algorithmus schon so komplex ist, daß seine angemessene systematische Konstruktion und Verifikation bislang noch aussteht, so ist er dennoch weit entfernt vom Umfang jener Art von Algorithmen, um die es in der Forschergruppe geht, beispielsweise in ihren drei Fallstudien, und die ohne softwaretechnische Methoden und Werkzeuge nicht beherrschbar sind. Dennoch nutzen verteilte Algorithmen der Forschergruppe, denn sie kommen in industrietypischen Anwendungen vor, verwoben mit anderen Komponenten. Beispiele dafür sind der Crosstalk-Algorithmus zur Synchronisation in Steuerungen, verschiedene Mutex-Algorithmen zur Vermeidung von Kollision, und Kommunikationsprotokolle aller Art. Es ist sicher interessant, die Fallstudien einmal systematisch nach diesen und anderen verteilten Basisalgorithmen abzusuchen. Dabei sollten verschiedene Muster der Einbettung verteilter Basisalgorithmen erkennbar werden. In ihrer bisherigen in [Rei97] dargestellten Form sind beispielsweise folgende Einbettungsformen sichtbar:

- über Transitionen: asynchroner Stack, Algorithmen zum wechselseitigen Ausschluß, Master/ Slave-Algorithmus.
- über Plätze: alternating bit- und sliding window-Protokoll, verteilter Request-service, verteiltes Sortieren (und andere constraint-Programme).
- mehrstufig: crosstalk $\rightarrow$ rearrangement (über Transitionen) und weiter in Umgebungen über Plätze.

Es gibt eine Reihe anderer Algorithmen, die sich keinesfalls so einfach in ihre Umgebung einbetten.

Unstrittig ist, daß systematisches Komponieren die Grenze des Beherrschbaren ein gutes Stück hinausschiebt. Damit wird aber noch keinesfalls jene Größenordnung erreicht, die in der Forschergruppe ansteht.

Literatur

[BA90] M. Ben-Ari. *Principles of Concurrent and Distributed Programming.* Prentice Hall international series in computer science. Prentice Hall, New York, 1990.

[Bar96] Valmir C. Barbosa. *An Introduction to Distributed Algorithms.* The MIT Press, Cambridge, Massachusetts, 1996.

[CM88] K. M. Chandy und J. Misra. *Parallel Program Design: A Foundation.* Addison-Wesley, 1988.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms.* Morgan Kaufmann, 1996.

[MP92] Zohar Manna und Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, 1992.

[MP95] Zohar Manna und Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Verification.* Springer-Verlag, 1995.

[Ray88] Michel Raynal. *Distributed Algorithms and Protocols.* Wiley series in computing. Wiley, 1988.

[Rei97] Wolfgang Reisig. *Elements of Distributed Algorithms: Modelling and Analysis with Petri Nets.* Erscheint demnächst, 1997.

[RH90] Michel Raynal und Jean-Michel Helary. *Synchronization and Control of Distributed Systems and Programms.* Wiley series in parallel computing. Wiley, 1990.

[Tel94] Gerard Tel. *Introduction to Distributed Algorithms.* Cambridge University Press, Cambridge, 1994.