Robust and accurate co-simulation master algorithms applied to FMI slaves with discontinuous signals using FMI 2.0 features

Andreas Nicolai[1]   Anne Paepcke[1]   Hauke Hirsch[1]

[1]Faculty of Architecture, Technische Universität Dresden, Germany, `andreas.nicolai@tu-dresden.de`

Abstract

Error control in system simulation using co-simulation techniques is a task for the employed simulation master. With the availability of the FMI standard version 2.0 and rollback capabilities of simulation slaves, master algorithms can be implemented with support of error controlled integration. Particularly, for automated integration tools, the problem-specific dynamic adjustment of communication interval lengths becomes a necessity to obtain reliable co-simulation results while maintaining calculation efficiency. The article discusses various master algorithms and time step adjustment strategies using a test case with discontinuous input/output signals. As expected, fixed-step Gauss-Jacobi and Gauss-Seidel algorithms are found to be generally unsuited for the task. Iteration-based time step adjustment rules are an improvement, yet cannot recognize discontinuities resulting from time-event. Since the traditional Richardson/step-doubling error estimate also fails to recognize discontinuous signal changes, a slope-based modified Richardson-test is introduced and successfully applied. Finally, it is concluded that a suitable master algorithm for such problems is the non-iterating Gauss-Seidel with modified Richardson communication interval adjustment.

Keywords: FMI, co-simulation, master algorithm, error control, adaptive

1 Introduction to FMI Co-Simulation

The FMI standard is an established simulation runtime coupling standard, implemented by many simulation tools, already. The standard is maintained by the Modelica Association and defines two operation modes; model exchange (single central integration core) and co-simulation (each slave has its own time integration engine). In this article we discuss the co-simulation approach. In this simulation coupling procedure, a simulation master requests simulation slaves to integrate a part of the simulation time interval, and communicate output variables to the master. Hence, this part is termed communication interval.

In the version 2.0 of the standard, a new feature was introduced that allows the master to trigger slaves to store their current state and restore it later. This permits the solver to rollback the state of an FMU to a previous state and repeat an already calculated communication interval. With that capability a co-simulation master has now several different options at obtaining a fully coupled/implicit solution of the coupled problem.

Typical choices for such co-simulation master algorithms have already been investiged (Clauß et al., 2017; Schierz et al., 2012). The authors also show that other new features of the FMI standard, like input/output extrapolation, positively influence simulation performance and stability. Further, communication step adjustment strategies were discussed that take into account the mathematical characteristics of the coupled simulation slaves.

In this article a special subset of problems will be analyzed: co-simulation slaves with discontinuous real signals.

1.1 Reference implementation MASTERSIM

The standard master algorithms discussed in aforementioned publications were implemented in an open-source reference implementation of a co-simulation master. The software MASTERSIM[1] has been used successfully in the scope of building energy performance and system simulation. In particular, the master was used for the co-simulation of a Modelica control model coupled to the building energy simulation model NANDRAD (Nicolai and Paepcke, 2017), whereby the building energy system FMU implemented already FMI v. 2.0 features (Nicolai and Paepcke, 2016).

The implemented algorithms are, however, of generic nature and can be applied to a wide range of physical problem domains. In many of such domains, discontinuous signals and variables appear and shall be treated correctly by the co-simulation master. With *correctly* we mean, that the simulation results are accurate within a requested/accepted tolerance band.

In order to analyze behavior of systems with discontinuous signals, we will use a dedicated test case to observe effects and results related to the chosen co-simulation master algorithms.

2 Test Case and Reference FMUs

The investigated test case was designed by C. Clauß (Clauß and Majetta, 2016):

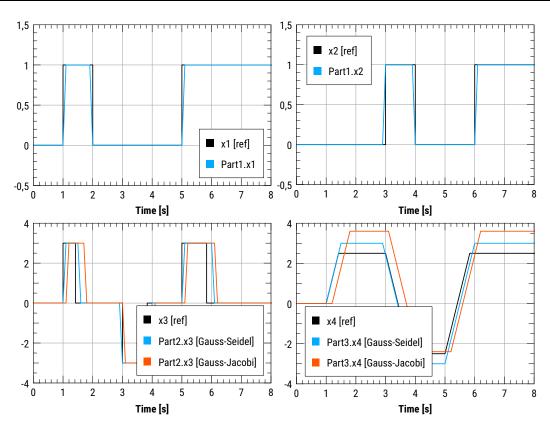[1]`https://sourceforge.net/p/mastersim`

Figure 1. Non-iterating, constant step, step size 0.1 s; Reference solution (black), GAUSS-JACOBI (red), GAUSS-SEIDEL (blue)

$$x_1 = \begin{cases} 0 & t < 1 \quad \text{or} \quad 2 \leq t < 5 \\ 1 & else \end{cases} \quad (1)$$

$$x_2 = \begin{cases} 0 & t < 3 \quad \text{or} \quad 4 \leq t < 6 \\ 1 & else \end{cases} \quad (2)$$

$$x_3 = \begin{cases} 3 & x_1 = 1 \quad \text{and} \quad x_2 < 0.01 \quad \text{and} \quad x_4 < 2.5 \\ -3 & x_1 < 0.001 \quad \text{and} \quad x_2 > 0 \quad \text{and} \quad x_4 > -2.5 \\ 0 & else \end{cases}$$

$$(3)$$

$$\dot{x}_4 = 2x_3 \quad (4)$$

The solution shall be obtained for the variables x_1, x_2, x_3, x_4 in the time interval $t \in [0, 10]$, with $x_4(0) = 0$. Note, that the conditions for variable x_3 are not formulated as comparisons with 1 or 0 for the variables x_1 and x_2, but instead allow for small tolerances that may appear through use of difference-quotient approximations in Newton algorithms[2].

2.1 Reference Solution

The problem has an exact solution, with time and state events at: $t = 1 + 2.5/6$, $t = 3 + 5/6$ and $t = 5 + 5/6$.

---

[2]Since the problem is linear, Newton algorithms are not meaningful and are not discussed in this article.

2.2 FMI Co-Simulation Scenario

The problem shall be split into 3 parts (corresponding to one FMU, each), where the first implements equations (1) and (2), the second implements eqn. (3) and the third implements eqn. (4). Only part 2 and 3 are coupled in a cycle. Part 1 shall be always evaluated first. In iterative algorithms for the cycle, part 2 shall be evaluated before part 3. Only part 3 needs to implement time integration, whereby an analytic solution is available.

2.3 FMI Slave Generation

Initially, the FMI slaves were generated by writing the equations in Modelica and exporting co-simulation slaves. However, small deviations from the requested solution appeared. For example, when FMU 1 was requested to take 20 communication steps with 0.1 s each, the end time point of the last interval passed to the FMU was 2.0000000000000004 s. However, the Modelica-exported FMU still returned $x_1 = 1$, either due to internal rounding errors or because the variable x_1 was evaluated at the beginning of the interval [1.9...2.0]. This was in disagreement with the formulated problem, so a direct implementation of the FMI slaves in C/C++ code was used instead, see also source code from of the MASTERSIM test suite (Nicolai, 2018b).
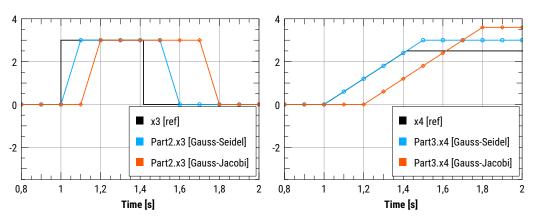
Figure 2. Non-iterating, constant step, step size 0.1 s, enlarged view; Reference solution (black), GAUSS-JACOBI (red), GAUSS-SEIDEL (blue)

3 Master Algorithms without Error Estimation

3.1 Constant-step Master Algorithm without Iteration (FMI v1)

Two popular examples for non-iterative constant-step co-simulation master algorithms are GAUSS-JACOBI and GAUSS-SEIDEL, both applicable for FMI slaves implementing only version 1 of the interface specification. With GAUSS-JACOBI, all FMUs are requested to integrate a time interval $[t, t+h]$ using input signals from time t. With GAUSS-SEIDEL, the FMUs are evaluated in sequence, where results from a previous FMU evaluation (output signals at $t+h$) are passed as inputs to a later FMU. Figure 1 shows results obtained when using constant communication interval lengths of 0.1 s. Clearly, the information propagation in the GAUSS-SEIDEL method leads to better results compared to GAUSS-JACOBI.

Results of FMU part 1 (variables x_1 and x_2) are identical for both methods, since they only depend on time. The differences in variables x_3 and x_4 are clearly visible. While in GAUSS-SEIDEL a change in variable x_1 or x_2 is already noted in the same communication interval, with GAUSS-JACOBI the information takes one more interval to reach FMU part 2 (variable x_3) and a further interval to influence variable x_4 (see Figure 2).

Of course, the accuracy of the solution can be increased by using smaller communication interval lengths, at the expense of computational (and mostly wasted) power.

3.2 Iterative Algorithm with Constant Steps

Once an FMU implements rollback capabilities, the co-simulation master can use iterative algorithms. It can be expected, that use of an iterative algorithm ensures information to propagate into all coupled FMUs in a cycle within the *same communication interval*. This implies, that the same communication interval is (re-)computed multiple times.

For iterative methods, a criterion for stopping the iteration is needed. In MASTERSIM the weighted-root-mean-square norm (5) is used for all exchanged real input/output variables. Variables of other types are ignored in the test.

$$|y|_{WRMS} = \frac{1}{n} \sqrt{\sum_{i=1}^{n} \left(\frac{y_{new,i} - y_{old,i}}{|y_{new,i}| r_{tol} + a_{tol}} \right)^2} \qquad (5)$$

Convergence is reached when $|y|_{WRMS} \leq 1$, with r_{tol} and a_{tol} being relative and absolute tolerances, respectively. In order to avoid stalling of iteration, a maximum number of iterations is specified. Once this limit is exceeded, the simulation continues with the results from the last iteration, which may be potentially inaccurate.

Indeed, an iterative algorithm used in conjunction with GAUSS-SEIDEL (and tolerances $r_{tol} = a_{tol} = 10^{-5}$), improves the results as shown in Figure 3.

3.3 Iterative Algorithm with Adaptive Communication Step Size

When encountering a state event[3], the GAUSS-SEIDEL method does not converge. This observation can be used to adapt the communication step size and thus detect the state event with a discontinuous first derivative reliably. Whenever an iteration does not converge within the given maximum number of iterations, the communication step is reduced and the step is reattempted. If the iteration converges, (in this test case with linear coupling usually after one repeated step,) the step size is enlarged. This procedure successfully captures the state event, i.e. the correlated variables x_3 and x_4 (Figure 4).

The factors, by which the communication step is reduced or enlarged, determine to a large extend the overall simulation effort. Also, a lower limit for the time step has to be specified, below which a step is accepted even without convergence (limit to fall-back to non-adaptive variant). Without such a criterion, the simulation will get stuck at the first state event by continuously reducing the

---

[3]A state event describes a discontinuity in a variable or its derivative resulting from a condition that depends on the value of the variable itself. In contrast to that, a time event indicates a similar discontinuity/condition that exclusively depends on the simulation time.
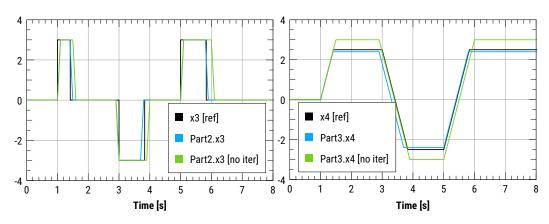
Figure 3. GAUSS-SEIDEL, iterating, max 2 iterationens, step size 0.1 s (blue), non-iterating GAUSS-SEIDEL (green)
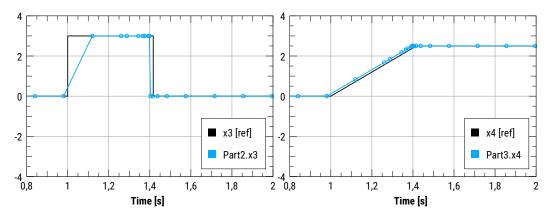

Figure 4. GAUSS-SEIDEL, iterating, max 2 iterations, variable step sizes (max. 0.14 s, min 0.005 s)

time step until rounding errors prevent further progress. Also, an upper limit of the time step is needed, to avoid overshooting over significant events. This limit is problem specific and for building simulation models, for example, the time step shall not exceed 30 min, so that hourly climate data is correctly considered. The heuristic parameters for this test case were set as follows: $f_{red} = 0.2, f_{exp} = 2, h_{fallback} = 0.005\,s, h_{max} = 0.14\,s$.

Consider Figure 4, where the occurance of the first state event is shown. We can observe the following:

- The time delay of the curves x_3 and x_4 results from the delayed jump of variable x_1 (occurs at time $t = 1.12s$).

- $t = 1.344s$ is reached with maximum step size of $0.14s$, the next step with same step size leads to $x_4 > 2.5$ and result in a convergence error.

- Hence, the step will be reduced by a factor of 5 ($\Delta t = 0.0224s$) and $t = 1.366s$ is reached; with $x_3 = 3$ and $x_4 < 2.5$ same as previous step and successful convergence, and step size is enlarged (doubled) to $\Delta t = 0.0448s$ afterwards.

- For the next interval, $0.0448s$ is again too long, a reduction of step size to $\Delta t = 0.00896s$ brings the solution even closer to the point of state event. Once

again the step will be doubled after iteration has converged.

- The interval $1.375s$ to $1.393s$ will be completed with exactly one iteration; conditions $x_3 = 3$ and $x_4 < 2.5$ remain unchanged.

- In the next interval, the state event is reached. First the step size is reduced to $\Delta t = 0.007168s$. With this time step, the state event is surpassed and the step is again reduced to $\Delta t = 0.0014336s$. Now, the step size is below the assigned lower limit of $0.005s$, and the step is processed/accepted without iteration.

- Afterwards the step is doubled to $\Delta t = 0.0028672s$, which is still below the fallback limit, and results are again accepted without iteration.

- Now, that the discontinuity has been surpassed, in the next intervals only one iteration is needed and the step sizes are doubled after the end of each interval (see Figure 5 for variation of communication step sizes).
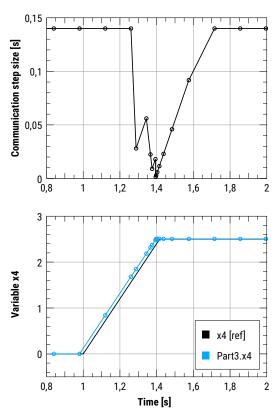
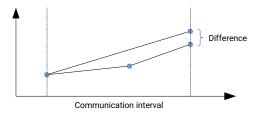**Figure 5.** Accepted step sizes and resultant variable x_4 for iterative GAUSS-SEIDEL method.

Figure 6. Illustration of Richardson/Step-doubling error test

4 Algorithm with Error Control

4.1 Error Estimate

While the previous algorithm did resolve the state event, it is not able to recognize discontinuities in variables that only depend on time (in this case, variables x_1 and x_2). Utilizing the rollback functionality of FMI v2.0, a straightforward approch is the use of Richardson-extrapolation (step-doubling) error test (Shampine, 1985). In the basic variant of this method, the step is executed first in full length, followed by a rollback to begin of the last step and execution of two steps with half communication length. The difference between the final results is an indicator for the integration error (Figure 6).

However, this error estimation fails when discontinuities appear within the interval, as in the case of variable x_1. In this case, the results of the full and the half-step are identical, and consequently the error cannot be detected.
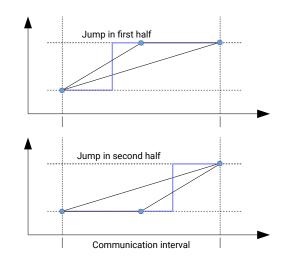

Figure 7. Inability of step-doubling/Richardson error test to recognize discontinuities in the solution; blue curve shows correct solution, the points show the results after FMU evaluation

Figure 7 illustrates the problem for both cases, when the discontinuity arises either before or after half of the interval.

A solution to this problem is to construct the error estimate based both on function results *and* slopes of the intervals, eqns. (6) and (7), respectively. Derivative information provided by FMUs themselves (a feature rarely supported, yet) is of no use, since derivatives at the end of the interval may well be the same for the full and a half interval (see Figure 7, bottom). Hence, the derivatives of the approximated solutions are constructed by difference-quotients:

$$\varepsilon_{Richardson} = \left| y(t+h)_{h/2} - y(t+h)_h \right| \tag{6}$$

$$\varepsilon_{slope} = h\left(\dot{y}(t+h)_h - \dot{y}(t+h)_{h/2} \right)$$
$$= h\left(\frac{y(t+h)_h - y(t)}{h} - \frac{y(t+h)_{h/2} - y\left(t+\frac{h}{2}\right)}{h/2} \right) \tag{7}$$

The value $y(t+h)_{h/2}$ is obtained after executing two half-steps and $y(t+h)_h$ is the value obtained with the one full step. The derivatives $\dot{y}(t+h)_h$ and $\dot{y}(t+h)_{h/2}$ at $t+h$ are approximated by backward finite differences using the full step and the second half-step, respectively. For the final error test, the worst-case of both error estimates is taken to determine the communication step size.

The derivation and formal analysis of the robustness of this error estimate is beyond the scope of this article. However, since the construction of the error estimate requires a full step and two half steps just as the Richardson-test, it is meaningful to always evaluate both error tests individually and take the more critical one. Hereby, the master algorithm will detect any discontinuity and adjust communication step sizes, accordingly.
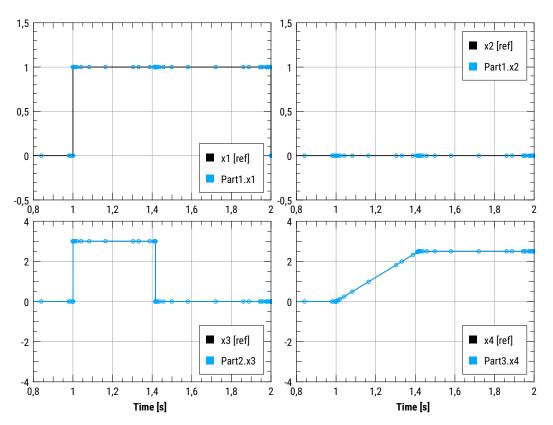
Figure 8. Results obtained with step-doubling/Richardson-error estimate and slope-difference error estimate

4.2 Results with Step Adjustment based on Richardson and Slope-Difference Error Estimates

The communication step is now adjusted based on the aforementioned error estimates, whereby the relative and absolute tolerances are set to 10^{-5}, each. The lower limit of a communication step is set to be $10^{-5} s$. This limit is also necessary for error controlled integration, since it is not possible to surpass a discontinuity otherwise. While implementing a formal root-finding procedure within the FMI co-simulation setup is possible with the FMI 2.0 roll-back functionality, it will hardly be efficient. Thus, there must be a mechanism implemented to disable the error check, once time steps become too small.

The computed results (see Figure 8) appear to be almost identical to the exact solution. Around the occurance of the first time and state events many evaluations close to the event's locations become visible. This is due to reduced step sizes because of error test failures.

After passing a discontinuity/event the communcation step is quickly enlarged again (with max. factor 2). The resulting characteristic spikes (see Figure 9) in the evolution of the communication step sizes are representative for linear or mostly linear problems with discontinuities.

The algorithm can reproduce the exact solution sufficiently well while maintaining the tolerance limit of max. 10^{-5} (see Figure 10). Apparently, the lower step size limit ($10^{-5}s$) was selected sufficiently small to avoid overshooting the tolerance band.
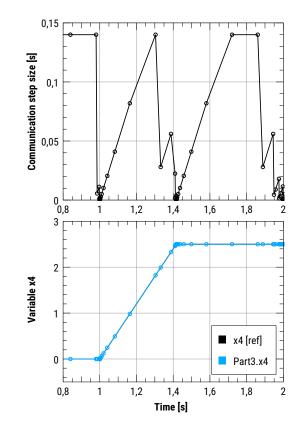

Figure 9. Accepted step sizes and resultant variable x_4 for the variant with error estimates.
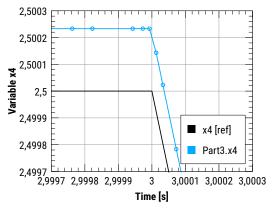
Figure 10. Remaining error in computed solution due to lower communication step limit

Further, the results are nearly the same for the iterative and non-iterative GAUSS-SEIDEL variants. The iterative variant typically requires two evaluations of each FMU per step. The simulation effort for the non-iterative variant is halved compared therewith.

5   Conclusions

Problems related to GAUSS-SEIDEL iteration methods in conjunction with discontinuities are well known for many decades, and presented stabilization techniques (iteration limit, lower time step limit, etc.) are state-of-the-art. Interestingly, though, these problems appear anew with the introduction of FMI for co-simulation, since authors/providers of individual FMUs may not have the global view necessary to identify such problems (or do not pay attention to potential numerical problems arising in the coupled simulation). Also, many straight-forward implementations of control models will generate discontinuous real signals, which should be handled in a robust way by co-simulation masters.

Without the FMI v. 2.0 state rollback feature, a sufficiently accurate solution of such problems cannot be guaranteed. Users of a co-simulation master would have to estimate a sufficiently small communication step, likely resulting in poor simulation performance. Still, the risk of unwanted errors remains.

In the presented case, a constant step size of 10^{-5}s would have been necessary for FMU v. 1 algorithms, resulting in 10^6 FMU evaluations. With the step adapting iterative GAUSS-SEIDEL algorithm, only 2639 FMU evaluations were needed. Hence, we can only recommend to implement FMI interface version 2.0 with rollback functionality (see open-source project FMICodeGenerator (Nicolai, 2018a) to assist in generating a minimal code base of such FMUs).

Implementing an error test procedure is generally advisable. However, when dealing with FMUs emitting discontinuous signals, a step-doubling error test method should be complemented with the presented slope-based check. Otherwise, the co-simulation master will not be able to re-liably detect discontinuities arising from time-events. For the presented test case, the variant without iteration and included error test yields the best performance, i.e. smallest number of FMU evaluations while maintaining accuracy. A more in-depth analysis of the various algorithms tested and the respective FMU evaluation counts can be found in the german test case publication (Nicolai, 2018c).

One has to keep in mind that the upcoming FMI 3.0 standard will address such issues to some extend. It will allow FMUs to detect discontinuities and return prematurely to the master and by this notify the master about the exact location of the discontinuity. However, similarly to the introduction of the FMI 2.0 standard, it will take several years until simulation slaves and co-simulation masters implement support for such FMI 3.0 features. Until then, the proposed algorithm and error detection method will be an effective way of ensuring accuracy.

Lastly, the test case shows how much influence the choice of master algorithm and heuristic parameters may have on the results. Also, during development of the MASTERSIM code, many bugs appeared that may involuntarily cause the coupled simulation to generate wrong results, despite the fact that the stand-alone tests done with test FMUs provided on fmi-standard.org ran successfully. We conclude, that co-simulation masters should additionally be tested with coupling scenarios and tested against provided, algorithm-specific results.

6   Acknowledgements

References

Christoph Clauß and Kristin Majetta. FMI Co-Simulation Test Scenarios. personal communication, 2016.

Christoph Clauß, Kristin Majetta, and Richard Meyer. Application of Richardson Extrapolation to the Co-Simulation of FMUs from Building Simulation. In *Proceedings of the 12th International Modelica Conference, Prague, Czech Republic,*, pages 79–88, 2017. doi:10.3384/ecp1713279.

Andreas Nicolai. FMI Code Generator project. Github Repository, 2018a. URL https://github.com/ghorwin/FMICodeGenerator.

Andreas Nicolai. Open-Source Co-Simulation Master MASTERSIM, 2018b. URL http://bauklimatik-dresden.de/mastersim.

Andreas Nicolai. Co-Simulations-Masteralgorithmen - Analyse und Details der Implementierung am Beispiel des Masterprogramms MASTERSIM. *Qucosa*, 2018c. doi:urn:nbn:de:bsz:14-qucosa2-319735. URL

`http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa2-319735`.

Andreas Nicolai and Anne Paepcke. Transformation der Gebäudeenergiesimulation NANDRAD mit variablem Zeitschrittlöser in eine Co-Simulation. In *Proceedings of the BauSIM 2016 in Dresden*, 2016.

Andreas Nicolai and Anne Paepcke. Co-Simulation between detailed building energy performance simulation and Modelica HVAC component models. In *Proceedings of the Modelica Conference 2017 in Dresden (DOI:10.3384/ecp17132)*, 2017.

Tom Schierz, Martin Arnold, and Christoph Clauß. Cosimulation with communication step size control in an FMI compatible master algorithm. In *Proceedings of the 9th International MODELICA Conference; September 3-5; 2012; Munich; Germany*, 2012.

L. F. Shampine. Local error estimation by doubling. *Computing*, 34(2):179–190, Jun 1985. ISSN 1436-5057. doi:10.1007/BF02259844. URL `https://doi.org/10.1007/BF02259844`.