C12 Spheres' Collision – SLX / LEDA

Analytical Simulation / Event-oriented Model

Simulator: SLX (**S**imulation **L**anguage with e**X**tensibility) is a compact system for developing discrete event simulations. SLX is object-based, but not object-oriented. Most important characteristics of SLX are its layered architecture and a syntax similar to that of C.

Model: In order to deal with this problem by a discrete event simulator we determine both the time to the next interaction and the two spheres involved in this collision. Our algorithm uses the relative velocities of neighbouring spheres to find the minimum among all possible future interaction times. Based on that we identify the spheres to execute the next collision and determine their resulting velocities as initial conditions for the next iteration step.

```
for (j = 1; j <= N-1; j++)
if (dv[j] > 0.0) dt[j] = dx[j] / dv[j];
else dt[j] = 1e+10;
k = 1; dtmin = dt[k];
for (j = 2; j <= N-1; j++)
     if (dt[j] < dtmin)
     { dtmin = dt[j]; k = j; }
```

LEDA [1] (**L**ibrary of **E**fficient **D**ata types and **A**lgorithms) is a software library which introduces the algebraic *real* data type to C++. The *reals* are the best approximation of the mathematical real numbers $\mathbb{R}$. They offer exact results for the operators +, -, *, /, the *k*-th root for any natural number *k* and for the comparative operators ==, !=, <, >, ≤, ≥.

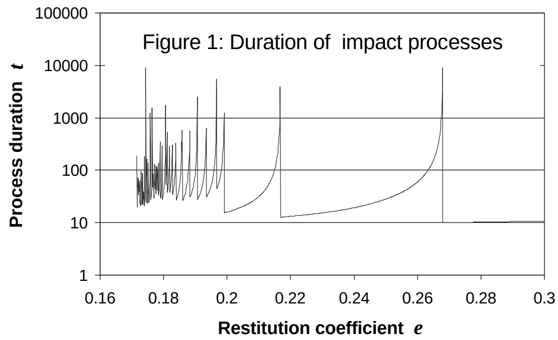
The *reals* "memorize" the calculations executed in an expression dag. Only if two *reals* are compared, or a number of the type *real* is to be output, the result is computed "as precise as necessary" in order to execute an exact comparison or in order to output the result with a given number of digits. This is however connected to a dramatically increasing runtime.

LEDA is coupled with SLX in order to increase the accuracy of the calculations essentially.

Task a: Modeling and simulating the impact pendulum by means of the simulation language SLX fully confirmed the results we already obtained using ACSL (polygon-type distance-time graph for e = 0.18, and final velocities $v_1 = v_2 = v_3 = 0$, $v_4 = 1$ (elastic case) and $v_1 = v_2 = v_3 = v_4 = 0.25$ (quasi-inelastic / plastic case)).

Additional examinations focus on the duration of the impact processes. The value of *e* also influences the duration of these which we define as the time up to reaching a state in which no further interactions will occur so that the terminal condition is met. A decrease in *e* shortens the duration of the process as long as this causes no additional interactions.

Additional interactions lead to a strong increase in the process duration. Fig. 1 shows this dependence.



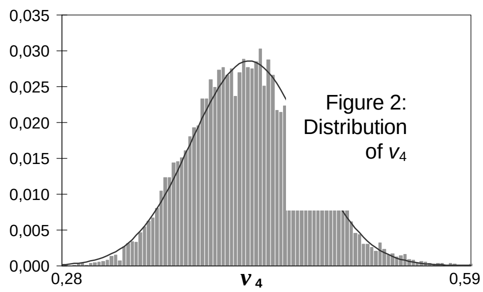
Task b1: Number of Collisions: With the use of the C++ library LEDA, it was possible to assure the numerically critical determination of the number of interactions *n* in the case of small restitution coefficients *e*. Some of the results are represented in Table 1.

e	0.1715735	0.1715730	0.1715729
n(*e*)	2947	6583	> 11216 *

Table 1: Number of interactions *n*(*e*)
(* after 2 months of comp. time, Sun Ultra 400MHz, 512 Mbyte RAM)

Results of **Task b2** and **Task c1** with SLX / LEDA are confirmed by former results e. g. with ACSL (Task b1: final velocities over *e*, and Task c1: e = 0.587401052 for boundary value problem $v_4 = v_0/2$).

Task c2: Distribution of v_4: With an again increased sample size of n = 10000 we obtain m = 0.423192, s = 0.042100, and CONF{0.422367 ≤ μ ≤ 0.424017}.



References

[1] *K. Mehlhorn and S. Näher. LEDA: A Platform for Combinatorial and Geometric Computing. Cambridge University Press, Cambridge, UK, 1999.*

Christian Gotzel, Rüdiger Hohmann, Carsten Pöge, Jörg Schwerdt, Otto-von-Guericke-Univ. Magdeburg
hohmann@isg.cs.uni-magdeburg.de