ul. Wolności 2, Blok 4/10
05-220 Zielonka
Polska

http://www.linkedin.com/in/tkoziara
t.koziara@gmail.com
Tel: +48 604 229 454

Tomasz Koziara

Employment

2016-now Contractor at Rullion, UK; remote.
Development and support of open-source computational mechanics software available at parmes.org in relation to its industrial application in the civil nuclear context in the UK. Funded by EDF Energy.

2012-2015 Lecturer in Computational/Theoretical Solid Mechanics; School of Engineering, Durham University, UK.
Work included multiple elements: research (individual research, consulting for industry, research grants), administrative (e.g. overseeing labs), hospitality (e.g. supporting open days), mentoring (e.g. supporting 4-7 final year students annually, 2 PhD students), and teaching (lecturing across years 2-4, to groups of 20-160 students).

2009-2011 Postdoctoral Research Fellow; School of Engineering, University of Glasgow, UK.
Development of a High Performance Computing multi-body dynamics analysis software, parmes.org/solfec-1.0, for applications to safety assessment of graphite nuclear plant cores. Funded by EDF Energy.

2007-2007 Part-time R&D engineer; Halcrow, Glasgow, UK.
Applying Diana, ANSYS and LS-DYNA Finite Element Method software packages for analysis of masonry, concrete and steel structures.

2001-2004 Software Developer; Robobat, Cracow, Poland.
Development of Finite Element Method software products in the Civil Engineering context.

Education

2004-2008 PhD in Computational Mechanics; School of Engineering, University of Glasgow.
Thesis "Aspects of Computational Contact Dynamics" received the ECCOMAS award as one of two best PhD theses of 2008 in Europe on computational methods in applied sciences and engineering. Also received the Zienkiewicz Prize in the UK in 2008.

1997-2002 Masters degree in Computational Mechanics; Department of Civil Engineering, Cracow University of Technology, Poland.
Thesis "XFEM modeling of cohesive fracture in concrete" defended with distinction. Based on originally implemented computational software.

Experience

- Software development
  - *Full-stack development:* I initially worked developing advanced Excel spreadsheets, mixing VisualBasic based frontends with C++ based COM component backends. Across the last 10+ years I developed an open-source High Performance Computing

SOLFEC-1.0 code (predominately written in C, using extended/embedded Python as an input interpreter, including an OpenGL GUI, MPI-based parallelism, and several output formats), documented it, and provided extended support for its industrial users.

- *Programming languages:* I predominately used C, C++ and Python to implement numerical software, as well as Scilab and MATLAB to prototype ideas across the past 20 years. I also used FORTRAN, VB, Julia, JavaScript, HTML, CSS.
- *Data structures and algorithms:* I implemented a variety of spatial search approaches (e.g. spatial hashing, segment-tree, kd-tree, octree, etc.) and classical computational geometry (e.g. Quickhull, GJK, etc.) and other classical algorithms and data structures (e.g. red-black trees, skip lists, memory pools, etc.). Many of these are included within SOLFEC-1.0 sources.
- *Parallelization:* SOLFEC-1.0 was a first code to fully parallelize the NSCD method using MPI. I have a breadth of experience of applying MPI to achieve distributed memory parallelism (e.g. non-blocking implementations of non-linear Gauss-Seidel and Newton solvers in SOLFEC-1.0, as well as complex load balancing therein; one-sided MPI-3.0 Remote Direct Memory Access based communication in SOLFEC-2.0). I developed a simple point-based load balancer, DYNLB, to optionally replace Zoltan in SOLFEC-1.0. I used various approaches for shared memory parallelism (e.g. OpenMP, ISPC native tasks, cpp-taskflow).
- *Vectorization:* I use an explicit SPMD on SIMD approach (Single Program Multiple Data on Single Instruction Multiple Data), ISPC, to achieve high efficiency on modern compute cores (e.g. I contributed the prefix sort example distributed with ISPC). PARMEC is an experimental computational code where I explore this programming paradigm as a primary design lens.
- *Visualization:* I have experience of visualizing geometrical data using OpenGL (e.g. SOLFEC-1.0's viewer).
- *Debugging:* Command line use of gdb/lldb on Linux/Unix/MacOS; VisualStudio debugger on Windows; TotalView in the context of MPI development.
- *Profiling:* MacOS Instruments/Time Profiler; gprof on Linux.
- *Documentation:* I developed parmes.org website and documentation using Sphinx and reStructuredText.
- *Testing:* SOLFEC-1.0 includes a set of custom Python-based automated non-regression tests related to the Validation Manual other validation work, as well as C-code level interactive tests of individual functionalities. SOLFEC-2.0 (an intended technological successor of SOLFEC-1.0) from the start incorporates Python unit-test based tests.
- *User support:* I have been actively supporting SOLFEC-1.0 and PARMEC users in recent years. This included aspects such as: communication (via GitHub issues, emails and face to face group discussions), development and improvements of documentation to better guide users, development of user requested features and computational approaches (e.g. XMDF export, Hybrid modeling) and functionality replication (e.g. LS-DYNA to PARMEC input file converter and replication in PARMEC of LS-DYNA's functionalities requested by industrial users, e.g. nonlinear springs), as well as bug fixing, and supporting the users directly on a dedicated HPC system.
- *Open-source:* See: https://github.com/parmes and https://github.com/tkoziara.

• *Research and development:* Due to my background in Computational Mechanics, I worked on a variety of mesh-based algorithms, mostly in the context of the Finite Element Method (both 2D and 3D, linear and nonlinear). In the context of SOLFEC-1.0, this included development of various aspects of implicit time-stepping methods for multi-body frictional contact/impact problems, e.g. non-smooth Newton methods for the frictional contact problem [1], time integrators for rigid rotations [2], and parallel code design

[3]. Most recently, I worked on a co-rotated and reduced order kinematic finite element model, suitable for the analysis of large scale multi-body structures in the non-smooth dynamics setting [4].

- *Reviewing:* see Researchgate profile; I acted in the role of reviewer for: International Journal for Numerical Methods in Engineering, Computer Methods in Applied Mechanics in Engineering.

- *Using proprietary FEA and CAD software:* I used Diana, ANSYS (for civil and mechanical engineering stress analysis applications), LS-DYNA (for drop test analysis) and used/taught ABAQUS FEA (3d, nonlinear, plasticity, buckling, contact), SolidWorks and AutoCAD.

- *Teaching:* As a postgraduate student at Cracow University of Technology I supported teaching of C programming to undergraduates. As a lecturer at Durham I taught: Year 2 Static Systems: matrix methods for statics of 2d trusses and frames (a lecture course for up to 160 students); developed a hands-on experimental laboratory to accompany this lecture course; taught Year 3 Civil Design: basics of concrete and steel design according to Eurocodes (a lecture course for 20-30 students of civil engineering); Year 4 Contact Mechanics: basics of classical contact mechanics and aspects of numerical contact analysis (a lecture course for 75-100 students; developed from scratch); Supported other teaching activities: Year 1 CAD (basics of SolidWorks), Year 3 Civil CAD (basics of AutoCAD), Year 4 ABAQUS FEA course (basics of nonlinear Finite Element Analysis).

- *Presenting:* I presented at 10+ international conferences. I gave a 2-part invited lecture on the HPC implementation of SOLFEC-1.0 during a summer school on Nonsmooth Contact Mechanics (Aussois, France, 2012), in consequence of an earlier invited stay at INRIA Grenoble.

- *Mentoring and advising:* as a lecturer at Durham I worked supporting students across a range of activities, from mentoring multiple groups of undergraduates across their 4-year university experience, through teaching in various modalities (e.g. as a lecturer, tutor, lab advisor, examiner), to supervising individual final year students. I co-supervised one PhD student (topic: fatigue analysis of wind turbine blades) and served as a primary advisor to another PhD student (topic: high performance contact detection in discrete element computations; this work completed after my departure from Durham).

This CV can be downloaded at https://parmes.org/_downloads/Tomasz-Koziara-CV-en.pdf for electronic use and access to embedded hyperlinks.

Selected publications

[1] T. Koziara, N. Bićanić. Semismooth Newton method for frictional contact between pseudo-rigid bodies. *Computer Methods in Applied Mechanics and Engineering* **2008**, *197*, 2763–2777.

[2] T. Koziara, N. Bićanić. Simple and efficient integration of rigid rotations suitable for constraint solvers. *Journal for Numerical Methods in Engineering* **2009**, *81*, 1073 – 1092.

[3] T. Koziara, N. Bićanić. A distributed memory parallel multibody Contact Dynamics code. *International Journal for Numerical Methods in Engineering* **2011**, *87*, 437–456.

[4] T. Koziara, S. Brasier, L. Kaczmarczyk. Co-rotated and reduced order finite element time integrators for multibody contact dynamics. *PARMES technical report TR1* **2017**.