A NEW PLATFORM FOR GLOBAL OPTIMIZATION*

C. Gong and Y. C. Chao
TRIUMF, 4004 Wesbrook Mall, Vancouver V6T 2A3, Canada

Abstract

A new software platform is described for the multi-objective global optimization of accelerator design. While local optimization is relatively simple, global optimization of accelerators remains a challenging task. The user often must write many lines of code to combine the output of a large variety of simulation engines. The optimization code also requires significant revision when applied to a different design. The TRIUMF optimization platform, based on the genetic algorithm, uses a flexible XML input format, in which users can easily combine multiple physics simulation engines in the same optimization problem. The platform is also parallel capable, designed to take advantage of high performance computation clusters. Capabilities of the platform are detailed and results of test problems and of the TRIUMF e-linac injector are shown.

INTRODUCTION

Simulation programs are indispensable for the design of modern accelerators. The many variables of an accelerator design can rarely be encompassed by a single simulation program. Often the design depends on results extracted from different programs. Another common scenario is to use the output of one code as the input of another [1], such as using GPT [2] to generate a beam from a gun and propagating the beam through Astra [3]. Global optimization is difficult due to the coupling between simulation programs. Much intermediate code is necessary to propagate variables between different programs and convert different input formats. In addition, the code is often written ad-hoc without serious attention paid to good software design and reusability. Testing and maintaining the intermediate code hinders the already complex task of creating a physically realistic simulation. The TRIUMF optimization platform is designed to mitigate these limitations.

DESIGN CRITERIA AND IMPLEMENTATION DETAILS

The principle goal of the platform is the global optimization of an accelerator design that requires the use of multiple simulation programs for each instance of the design. The platform mediates all interactions between the programs, including propagation of variables and extraction of constraint and objective values. The platform has the flexibility to support a *topology* composed of an arbitrary number of different simulation codes arranged in sequential and/or parallel order. While the primary purpose is the eventual global optimization of the TRIUMF e-linac [4], the software design is generic and can handle simulation programs of any nature.

The software is based on A Platform and Programming Language Independent Interface for Search Algorithms (PISA) [5] and Alternative Platform and Programming Language Independent Interface for Search Algorithms (APISA) [6], which use genetic optimization [7]. The scope of the platform is ambitious, thus has been subjected to rigorous software engineering methods in an effort for it to be maintainable and extensible. The platform is designed and written in full object-oriented C++ for Linux systems, with a focus on documentation and good coding convention.

One primary design goal of the optimization engine is to take advantage of parallel computing. The engine is designed and tested to work with WestGrid [8], a high performance computing network for Canadian institutions.

No restrictions are made on the type of decision variables, constraints, or objectives allowed, thus they can be any arbitrary numerical properties of the system. The platform allows the user to provide custom Python code to parse the simulation output and extract parameters of interest. Some common functionalities are provided in libraries, e.g., extracting beam emittance from Astra.

XML was chosen as the input file format for its standardization, flexibility, and extensibility. An input file consists of XML blocks that define the optimization *settings*, *variables*, *constraints*, and *objectives*. A *topology* block defines the simulation programs used and their dependencies, with each entry in the topology referred to as a *vertex*. An Astra vertex that depends on a GPT vertex runs in serial, while two vertices that are independent can run in parallel. A *units* block can also be defined to ease the transition between programs that have different unit conventions.

The platform features a sophisticated error handling scheme and gracefully works around any exceptions thrown by a simulation program (e.g., program hangs) or the operating system (e.g., *fork()* error), to prevent a single exception from destroying the entire run.

TESTING AND VALIDATION

The problem ZDT6 [9] was used to test the platform's capability in handling multi-vertex topologies (Fig. 1). ZDT6 is defined as

---

* Work supported by Natural Sciences and Engineering Research Council of Canada and National Research Council of Canada

$$\text{minimize } f_1(x) = 1 - e^{-4x_1} \cdot \sin^6(6\pi x_1)$$

$$\text{minimize } f_2(x) = 1 - \left(\frac{f_1(x)}{g(x)}\right)^2$$

$$g(x) = 1 + 9 \cdot \left(\frac{1}{9}\sum_{i=2}^{10} x_i\right)^{0.25}$$

subject to $0 \leq x_i \leq 1, i = 1, ..., 10$.

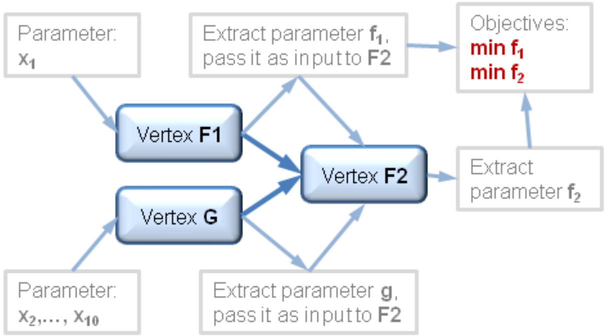

Figure 1: The ZDT6 test problem is modeled as three vertices: G, $F1$, and $F2$. G and $F1$ can be executed in parallel. $F2$ executes in serial with respect to G and $F1$, because $F2$ depends on the output of the other two. The parameter f_1 is used as both an input for $F2$ and also as an objective variable.

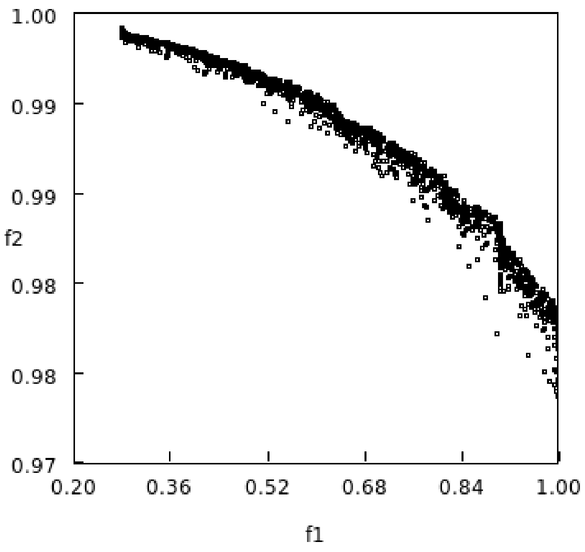

Figure 2: The pareto front shows the trade-off between the ZDT6 objective functions, f_1 and f_2.

ZDT6 has a non-convex solution space. Each vertex was written as a separate program, emulating a multi-engine simulation. For each *individual*, random numbers between 0 and 1 are assigned for each of the ten decision variables x_i, $i = 1, ..., 10$. x_1 is used as the input for the vertex $F1$, and the others for G. Non-decision variables $f1$ and g are extracted from $F1$ and G, respectively, and are then used as inputs for vertex $F2$, which outputs variable f_2. The objectives are minimizing f_1 and f_2. The resulting pareto front (Fig. 2) matches the results obtained by [10].

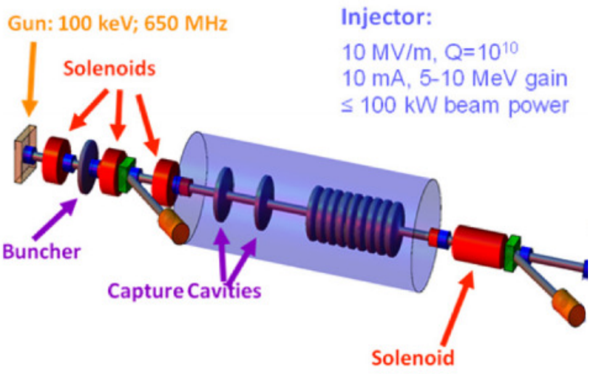

Figure 3: E-linac injector. Variables include B-fields for the solenoids and phases and E-fields for the cavities (buncher, two capture cavities, and 9-cell). Figure courtesy of [11].

The optimization platform was applied to an Astra model of the TRIUMF e-linac injector (Fig. 3), in which an initial 100 keV bunch is accelerated to 10 MeV. The simulation topology only consists of a single Astra vertex, but is a good test to determine whether the platform produces physically realistic results. The decision variables include the solenoid fields, the buncher and capture cavity fields and phases, and the 9-cell phase. Fig. 4 shows a pareto front that demonstrates a trade-off between transverse and longitudinal emittances. The results (Fig. 4, 5) are reasonable within the requirements of the e-linac and gives confidence to the optimization platform.

CONCLUSION AND FUTURE APPLICATIONS

The optimization platform has been implemented and shows good handling of arbitrary problem topologies in a parallel-capable environment. The platform is currently being used to study injector dynamics, and other near future applications are envisioned for TRIUMF. Ultimately the goal of the platform is the global optimization of the e-linac, a 50 MeV, 0.5 MW machine for photo-fission. Space is reserved for the future upgrade to an energy recovery linac, making possible simultaneous dual beam operation, with one beam used for rare isotope production and the other for FEL operation. The flexibility and scalability of the optimization platform is designed to prepare for such a complex task (Fig. 6).

REFERENCES

[1] F. Jones, Y. C. Chao and C. Gong, "Multi-Code Modelling of Momentum Collimation in the TRIUMF-ARIEL Linac," IPAC12, these proceedings.

[2] General Particle Tracer - `http://www.pulsar.nl/gpt/`

ISBN 978-3-95450-115-1

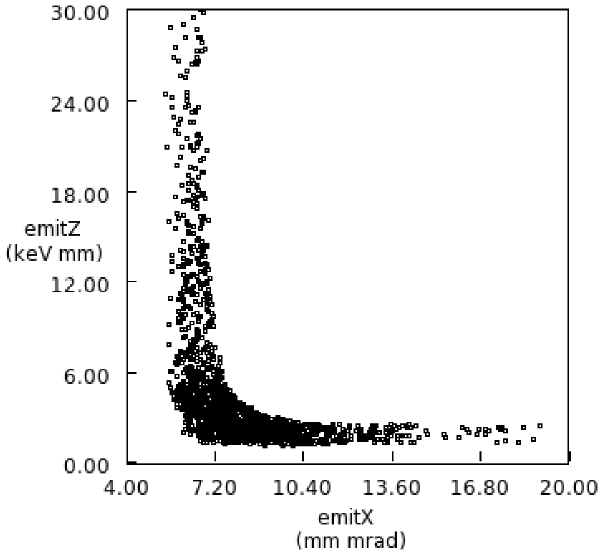

Figure 4: Optimization of the e-linac injector using Astra. The pareto front shows the trade-off in the designs between the two objectives, ϵ_x and ϵ_z. The optimization consists of only the Astra vertex in the topology, but shows the validity of the platform when applied to a real problem.

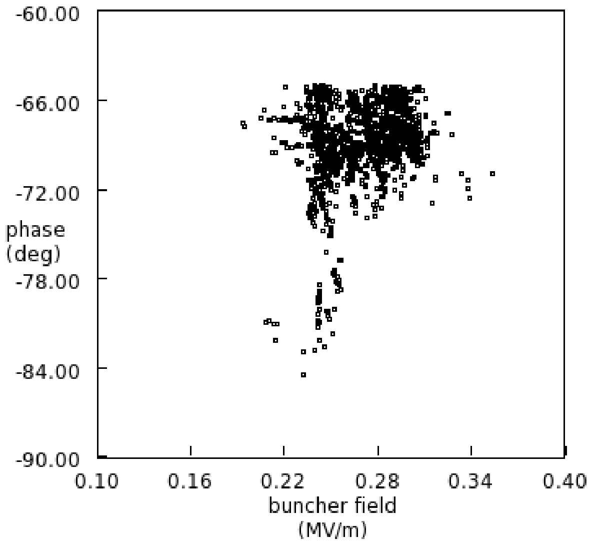

Figure 5: Optimization of the e-linac injector shows the optimal field for the first solenoid is 0.28 ± 0.04 MV/m. The phase represents the initial buncher phase at the beginning of the Astra run. The search range is between 0 and 3 MV/m.

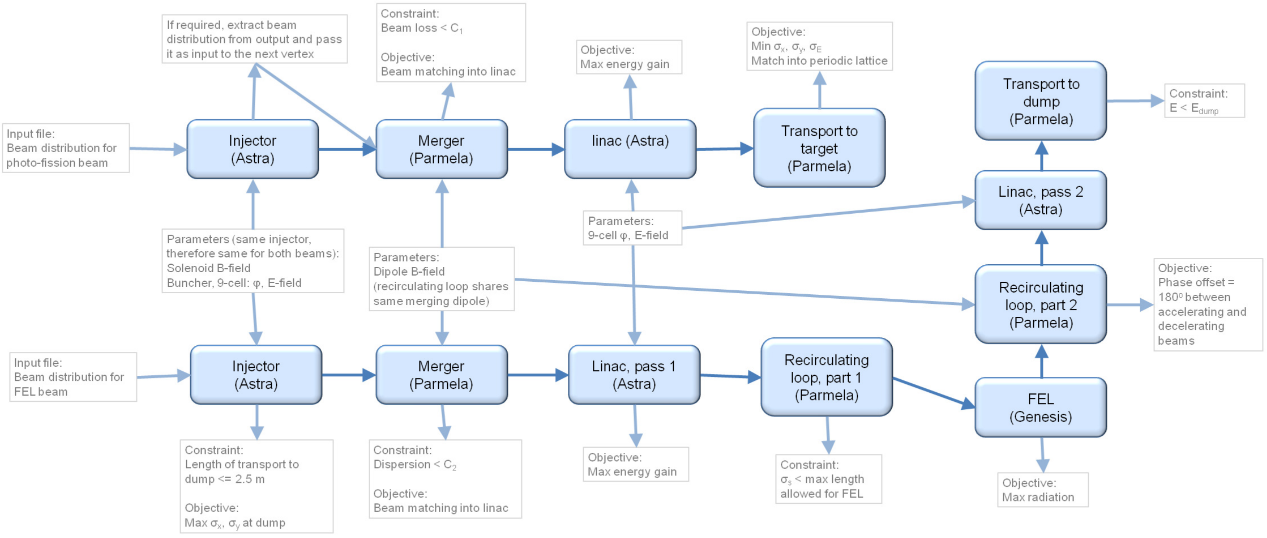

Figure 6: The many variables involved in the e-linac cannot be modeled by a single simulation program. The optimization framework is designed to handle the optimization of such systems.

[3] ASTRA - `http://tesla.desy.de/~lfroehli/astra/`

[4] S. Koscielniak et al., “ARIEL and the TRIUMF E-Linac Initiative, a 1/2-MW Electron Linac for Rare Isotope Beam Production,” LINAC08, Victoria, 2008.

[5] PISA - `http://www.tik.ee.ethz.ch/pisa/`

[6] APISA - `http://www.lepp.cornell.edu/~ib38/apisa/`

[7] E. Zitzler and M. Laumanns and L. Thiele, “SPEA2: Improving the Strength Pareto Evolutionary Algorithm.” Technical report, 2001.

[8] WestGrid - `http://www.westgrid.ca/`

[9] E. Zitzler and K. Deb and L. Thiele, “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results,” Evolutionary Computation, 8(2):173-195, 2000.

[10] M. Jiang and K. Zhu, “Multiobjective optimization by Artificial Fish Swarm Algorithm,”2011 IEEE International Conference on Computer Science and Automation Engineering , (CSAE), volume 3, pages 506-511, June 2011.

[11] Y .C .Chao et al .,“TRIUMF -VECC Electron Linac Beam Dynamics Optimization,” ICAP09, San Francisco, 2009.