Jinkō: Building a cloud simulation system for reproducible and multi-level cached in-silico trials

jinkō
'artificial' (人工, jinkō)
'population' (人口, jinkō)

VPH2022 virtual physiological human conference

Y. Parès[1], E. Tixier[1], M. Coudron[1], F. Cogny[1]
[1] Novadiscovery, Lyon, France
Contact: yves.pares@novadiscovery.com

BACKGROUND

What are you needs when building *in silico* clinical trials? We present here the core ideas and features of Jinko.ai, a web platform currently developed by Novadiscovery for trialists and QSP (quantitative systems pharmacology) modelers. Figure 1 shows which demands of these professionals Jinko.ai responds to. We mostly describe here the platform through the lens of the second category: Trial Design & Simulation.

In silico trials involve running many simulations on the same computational systems biology model, only varying the parameters and initial conditions. In this framework, we can identify a unitary brick of computation, which we call "atomic job": solving a specific version of a specific model using a specific set of parameters and initial conditions. An important thing to note is that this atomic job is deterministic: running it twice produces the exact same results. This property provides reproducibility, a key aspect of the scientific method. We can use that atomic job as part of many simulations and parameter-space exploration jobs, which are most of the time highly parallelizable. These atomic jobs being deterministic, these high-level jobs can share results and computation time.

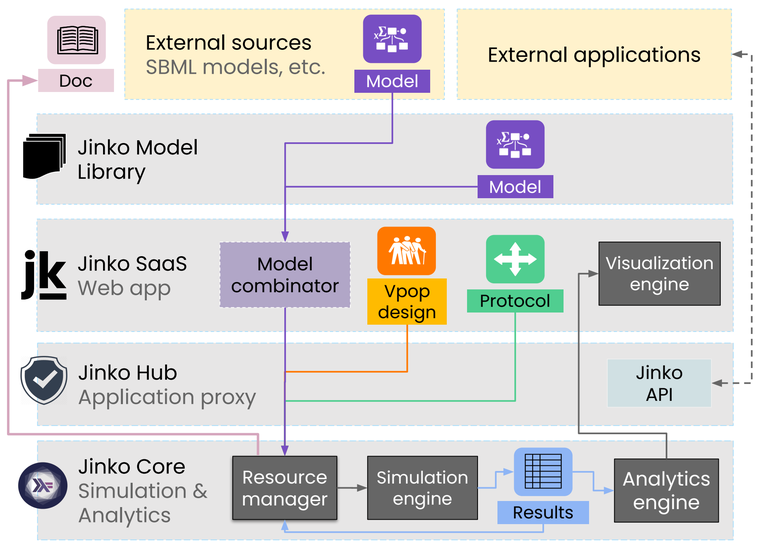

Figure 2: Trial Design & Simulation high level architecture and dataflow

TECHNICAL SUMMARY

Global architecture

Figure 2 shows the layers of Jinko.ai:
- a library of pre-existing models
- a web application in which trials are designed & submitted and trial results are visualized
- a simulation & analytics engine
- an application proxy, which links the previous two layers together and enables programmatic access to most of Jinko's features through an API

In silico simulation jobs

An atomic job in the context of *in silico* trials consists in solving the differential system derived from a biochemical model and serialising the outputs, typically time series of some species and parameters and scalar quantities of interest derived from them. For computational efficiency, the differential system is optimized using LLVM [1] and solved using Sundials' CVODE [2] library.

Job distribution system

The distributed simulation engine of the Jinko.ai platform is made to provide at-least-once semantics. It is based around three key ideas: (a) a fully symmetric and failure-tolerant "job-borrowing" mechanism that only needs a basic queue system and a relational database, (b) a representation of the various workloads as a hierarchy of deterministic, cached jobs [3], and (c) content-addressing of input resources, job instances and job results [4].

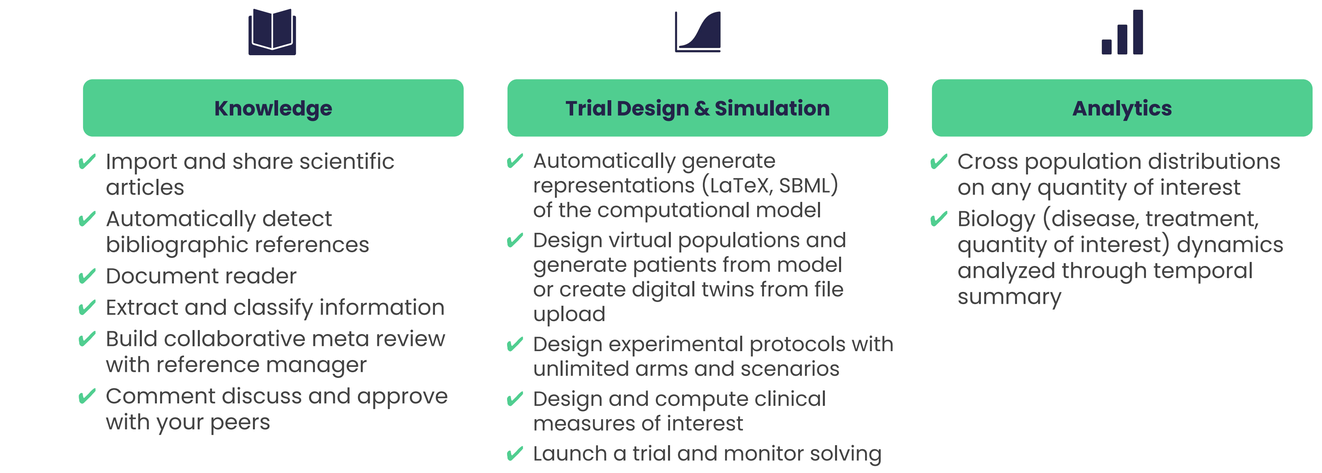

Figure 1: The three main Jinko modules, supporting an extensive part of any systems biology modeling project

METHODS

Modeling a clinical trial

Figure 2 shows the life cycle of the main entities around which jinko.ai is architectured.

- First we have the *inputs* that model a trial, and can be either designed in the platform or imported from external sources:

Model — A set of chemical equations, Ordinary Differential Equations, events through time (injections, meals, etc.) and scoring functions. Describes the dynamics of the constructs (variables to solve, parameters) at play in a specific biological system in the context of a given treatment, and how to derive numerical scores from the evolution of these quantities once simulated, to compute some measures of interest. Models are either imported from an external source, loaded from the Jinko.ai model library, or obtained through a combination of several other models.

Vpop design — *Virtual population design*, from which we can sample *virtual patients*. One single virtual patient is a set of specific values for some of the constructs of a model that should be fixed ahead of simulation. These values are either constants through time, time-varying functions or initial conditions of the system. Patients can also be created fully manually or imported from actual clinical trials (to produce *digital twins*).

Protocol — A set of different scenario cases that will be used to alter each virtual patient on the fly before running the simulation, describing the various arms of the trial. This methodology enables us to potentially re-use each patient with variations in each arm, therefore to have each patient be its own control.

- Prior to simulating, we can extract the following from our trial inputs:

Doc — A documentation automatically derived from the code. Different serialization formats are supported, including Markdown, LaTeX and SBML. This means a documentation which is exact by construction and always up-to-date, and which can be used either for internal purposes or directly submitted to regulatory agencies.

- Once the simulation of the trial is complete, these are the outputs we obtain:

Results — A set of time series and scalar measures (numerical scores, aggregates through time) for each patient, resulting from the simulation of one model over one virtual population and one protocol.

Running *in silico* simulations

- The simulation part of the software is validated using the method of manufactured solutions [5]. Calculation verification is carried out by simulating the differential system with different solver tolerance values until a satisfactory precision/cost tradeoff is achieved. From an end user perspective this means obtaining precise simulation results with little to no need to tweak intricate solver settings.

- Designing, running and inspecting trials is the main functionality of the Jinko platform, but Jinko will grow to accommodate the need for more complex schemes, such as resource-intensive post-simulation analytics, and calibrations. In the general sense, a calibration can be viewed as running repeatedly the same trial, but with alterations made between each iteration, in order to optimize some model or population parameters. Many classes of algorithms can be of use here (e.g bayesian sampling, genetic algorithms), some better suited to different scenarios or types of models. To accommodate for our current needs and enable easy experimentation and extensibility, the job distribution mechanism of the simulation & analytics engine has been designed with modularity and reusability in mind. This implies that different trials or calibrations will share their internal computations and results as much as possible, for instance if the same patient exists in two different populations or if two different protocol arms produce sometimes identical model overrides. From an end user perspective this means potentially providing trial results faster, especially when constructing populations or protocols as variants of previously used populations and protocols.

- Given the platform aims at serving various clients at the same time, it is important to ensure fairness when attributing computation time to all users. Besides automated scaling which is expected of this kind of cloud service, the platform uses a stochastic scheduling algorithm instead of simply relying on a job queue, to ensure that all trials currently being simulated progress at the same rate. From an end user perspective this implies that even if the platform is under heavy load, a user who submits a light trial (with only a few patients and protocol arms) will obtain a result quickly.

REFERENCES

1. Lattner C et al, "LLVM: A compilation framework for lifelong program..." Int Symposium on Code Gen and Opt, IEEE, 2004.
2. Hindmarsh AC et al., ACM Trans on Math Soft (TOMS) 31.3 (2005): 363-396.
3. Parès Y et al, "Composing effects into tasks and workflows." Proc 13th ACM SIGPLAN International Symposium on Haskell. 2020.
4. Dolstra E: "The purely functional software deployment model", PhD th, Utrecht U, 2006
5. Roache PJ, "Verification and validation in computational science and engineering", 895, Hermosa Albuquerque, NM, 1998.

www.novadiscovery.com/try-jinko