Fast New Methods for Solving Large Sets of Coupled Differential Equations at Scale in Scientific Applications

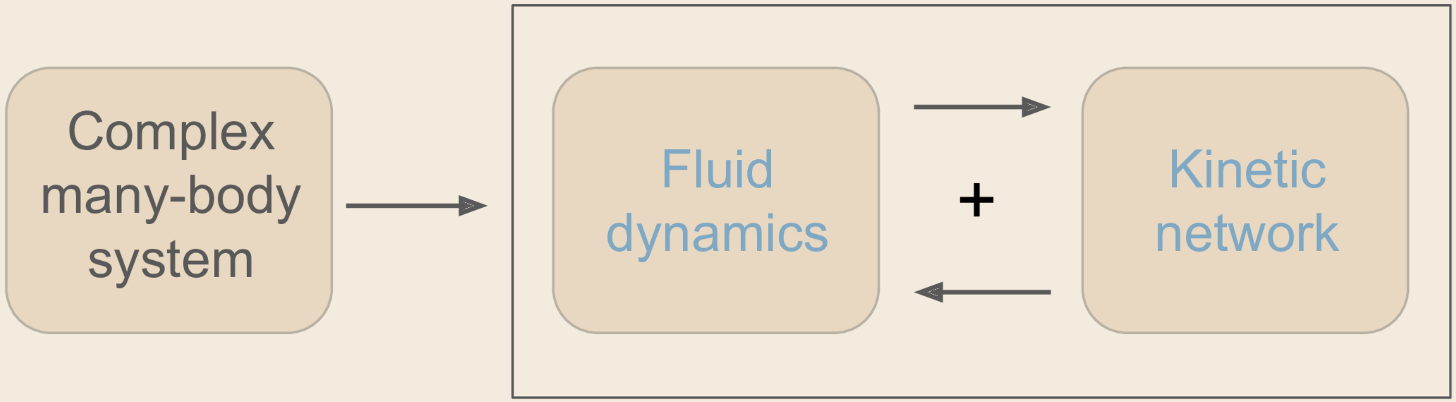
Mike Guidry

In collaboration with

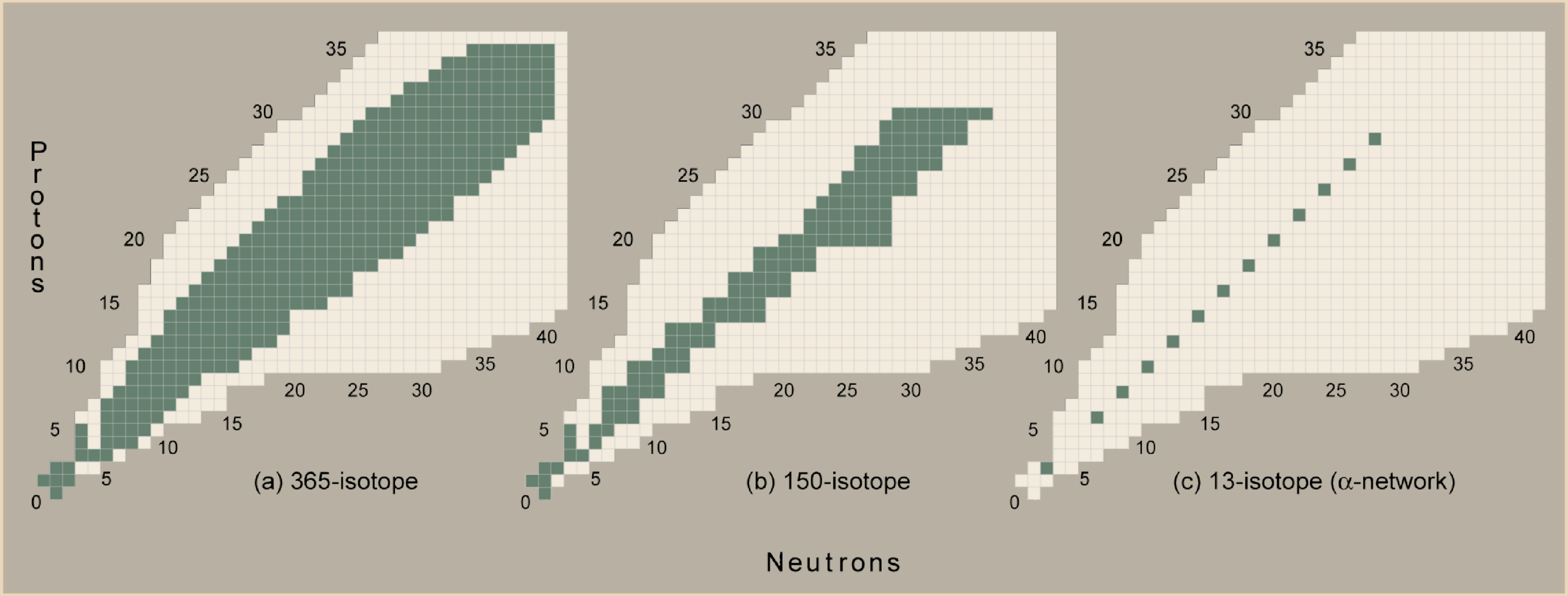
Azzam Haidar, Ben Brock, Daniel Shyles,
Stan Tomov, Jay Billings, and Andrew Belt

Fluid Dynamics Plus Kinetics Approximation

Many physical systems can be modeled by a fluid dynamics plus kinetics approximation.



Realistic Networks for the Type Ia Problem



<Network demo>

Coupling Realistic Thermonuclear Networks to Hydrodynamics

To incorporate realistic networks in astrophysical simulations we must improve (substantially) the speed and efficiency for computing kinetic networks coupled to fluid dynamics.

There are two general approaches that we might take:

- Improve the algorithms used to solve the kinetic networks.
- Improve the hardware on which the algorithms are executed.

This presentation is about using both to affect a dramatic improvement in the speed and efficiency for solving this problem.

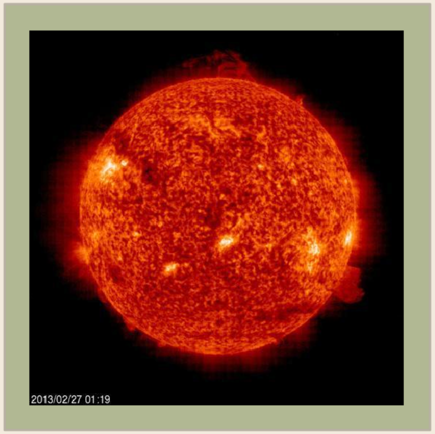
The Nature of Stiff Equations

- Stiff systems have two or more numerical timescales *differing by many orders of magnitude.*

- One quantitative measure of stiffness is the *ratio of the fastest and slowest timescales* in the system.

- Since most physical systems involve processes operating on very different timescales, *realistic problems tend to be at least moderately stiff.*

- Some stiff systems, such as those encountered in astrophysical thermonuclear networks, are *extremely stiff,* with timescales differing by *10-20 orders of magnitude.*



Methods to Integrate Stiff Equations

- There are two general approaches that we might use to deal with stiffness.

  - ***The traditional way:*** Integrate equations implicitly, which is stable but requires an iterative solution with matrix inversions at each step (expensive for large networks).

  - ***A new way:*** Replace equations with some that are more stable and integrate them explicitly.

- If we could stabilize explicit integration we could do each timestep more quickly in large networks.

Fundamental Sources of Stiffness

$$
\begin{aligned}
\frac{dy_i}{dt} &= F_i^+ - F_i^- \\
&= (f_1^+ + f_2^+ + \dots)_i - (f_1^- + f_2^- + \dots)_i \\
&= (f_1^+ - f_1^-)_i + (f_2^+ - f_2^-)_i + \dots = \sum_j (f_j^+ - f_j^-)_i
\end{aligned}
$$

The key to stabilizing explicit integration is to understand the three basic sources of stiffness for a typical reaction network:

- *Negative populations,*
- *Macroscopic equilibration*
- *Microscopic equilibration.*

Fundamental Sources of Stiffness

Negative populations

Macroscopic equilibration

$$
\begin{aligned}
\frac{d\boxed{y_i}}{dt} &= \boxed{F_i^+ - F_i^-} \\
&= (f_1^+ + f_2^+ + \ldots)_i - (f_1^- + f_2^- + \ldots)_i \\
&= (f_1^+ - f_1^-)_i + (f_2^+ - f_2^-)_i + \ldots = \sum_j \boxed{(f_j^+ - f_j^-)_i}
\end{aligned}
$$

Microscopic equilibration

The key to stabilizing explicit integration is to understand the three basic sources of stiffness for a typical reaction network:

- *Negative populations,*
- *Macroscopic equilibration*
- *Microscopic equilibration.*

A Simple Example

Species $\{\alpha, {}^{12}\mathrm{C}, {}^{16}\mathrm{O}\}$

Reactions $0:\ 3\alpha \rightleftarrows {}^{12}\mathrm{C} \qquad 1:\ \alpha + {}^{12}\mathrm{C} \rightleftarrows {}^{16}\mathrm{O}$

Network

$$
\begin{aligned}
\dot{Y}_\alpha &= -k_\mathrm{f}^{(0)} Y_\alpha^3 + k_\mathrm{r}^{(0)} Y_{12} - k_\mathrm{f}^{(1)} Y_\alpha Y_{12} + k_\mathrm{r}^{(1)} Y_{16} \\
\dot{Y}_{12} &= k_\mathrm{f}^{(0)} Y_\alpha^3 - k_\mathrm{r}^{(0)} Y_{12} - k_\mathrm{f}^{(1)} Y_\alpha Y_{12} + k_\mathrm{r}^{(1)} Y_{16} \\
\dot{Y}_{16} &= k_\mathrm{r}^{(1)} Y_\alpha Y_{12} - k_\mathrm{r}^{(1)} Y_{16}
\end{aligned}
$$

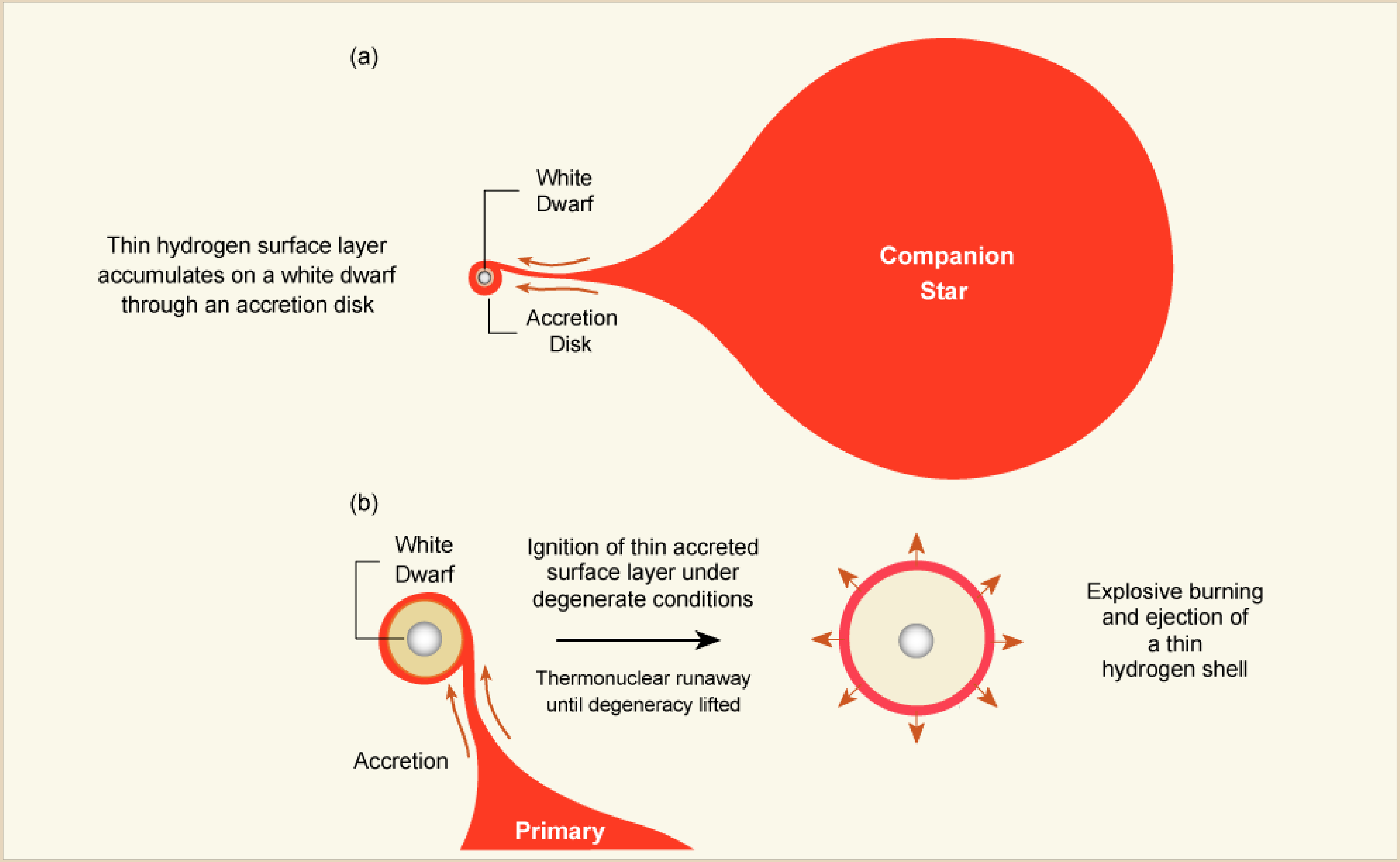
Macroscopic Equilibration Example:

$$
\dot{Y}_{12} = \underbrace{k_\mathrm{f}^{(0)} Y_\alpha^3 + k_\mathrm{r}^{(1)} Y_{16}}_{F^+} \ \underbrace{- k_\mathrm{r}^{(0)} Y_{12} - k_\mathrm{f}^{(1)} Y_\alpha Y_{12}}_{F^-}
$$

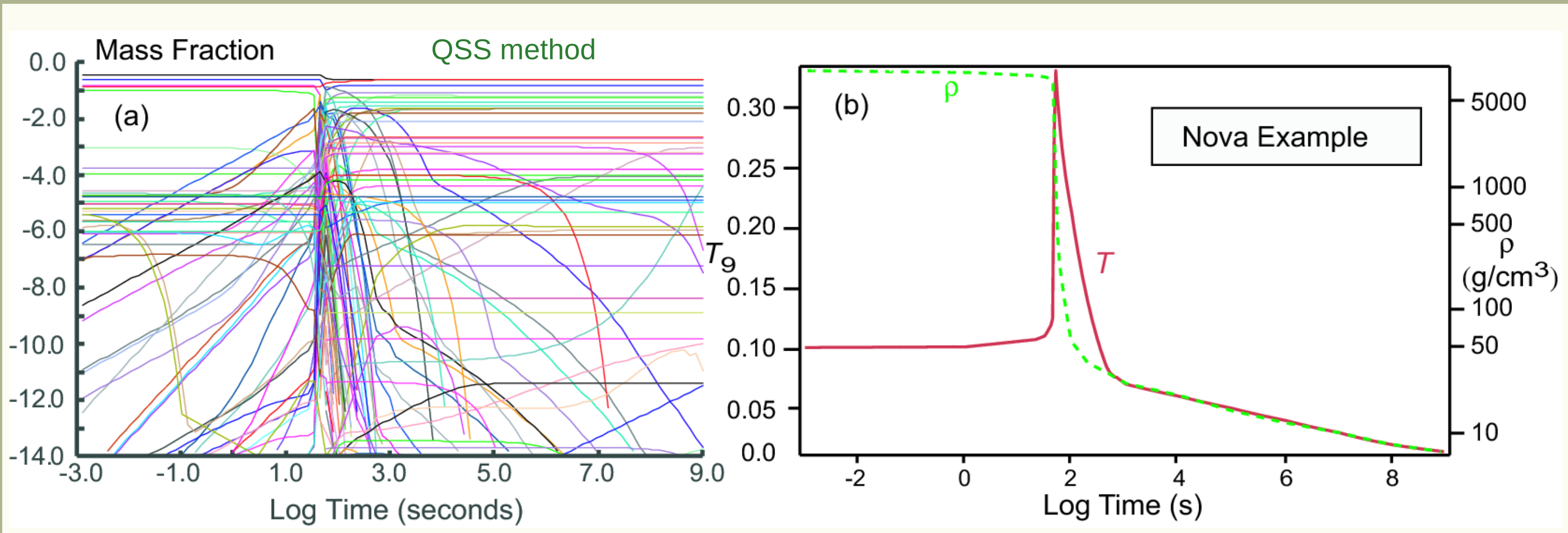
Microscopic Equilibration Example:

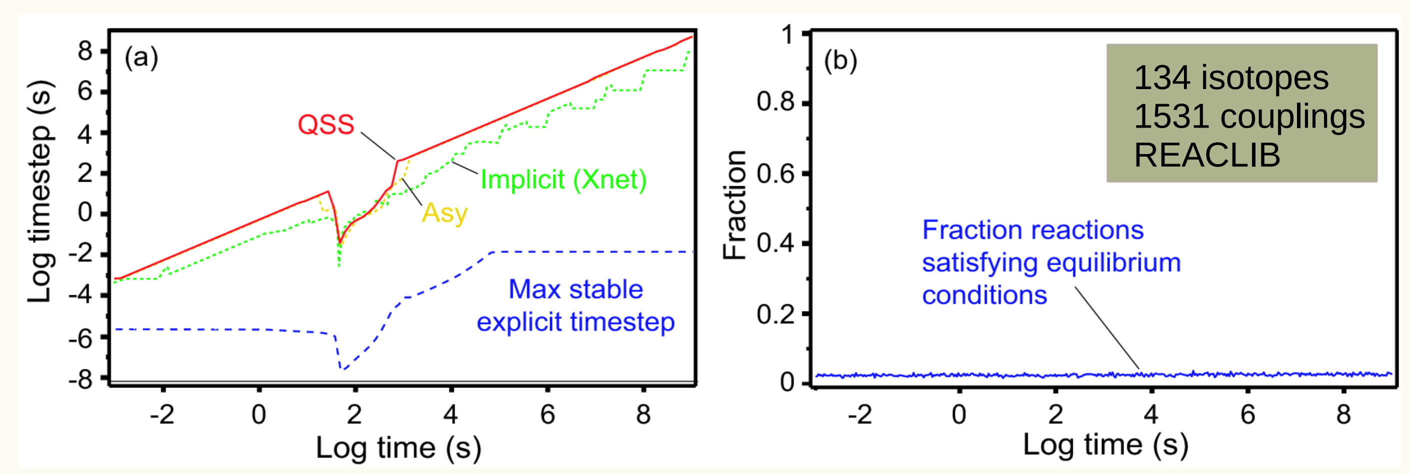
$$
\dot{Y}_{12} = \underbrace{k_\mathrm{f}^{(0)} Y_\alpha^3 - k_\mathrm{r}^{(0)} Y_{12}}_{3\alpha \rightleftarrows {}^{12}\mathrm{C}} \ \underbrace{+ k_\mathrm{r}^{(1)} Y_{16} - k_\mathrm{f}^{(1)} Y_\alpha Y_{12}}_{\alpha + {}^{12}\mathrm{C} \rightleftarrows {}^{16}\mathrm{O}}
$$

Example: Simulations of Nova Explosions



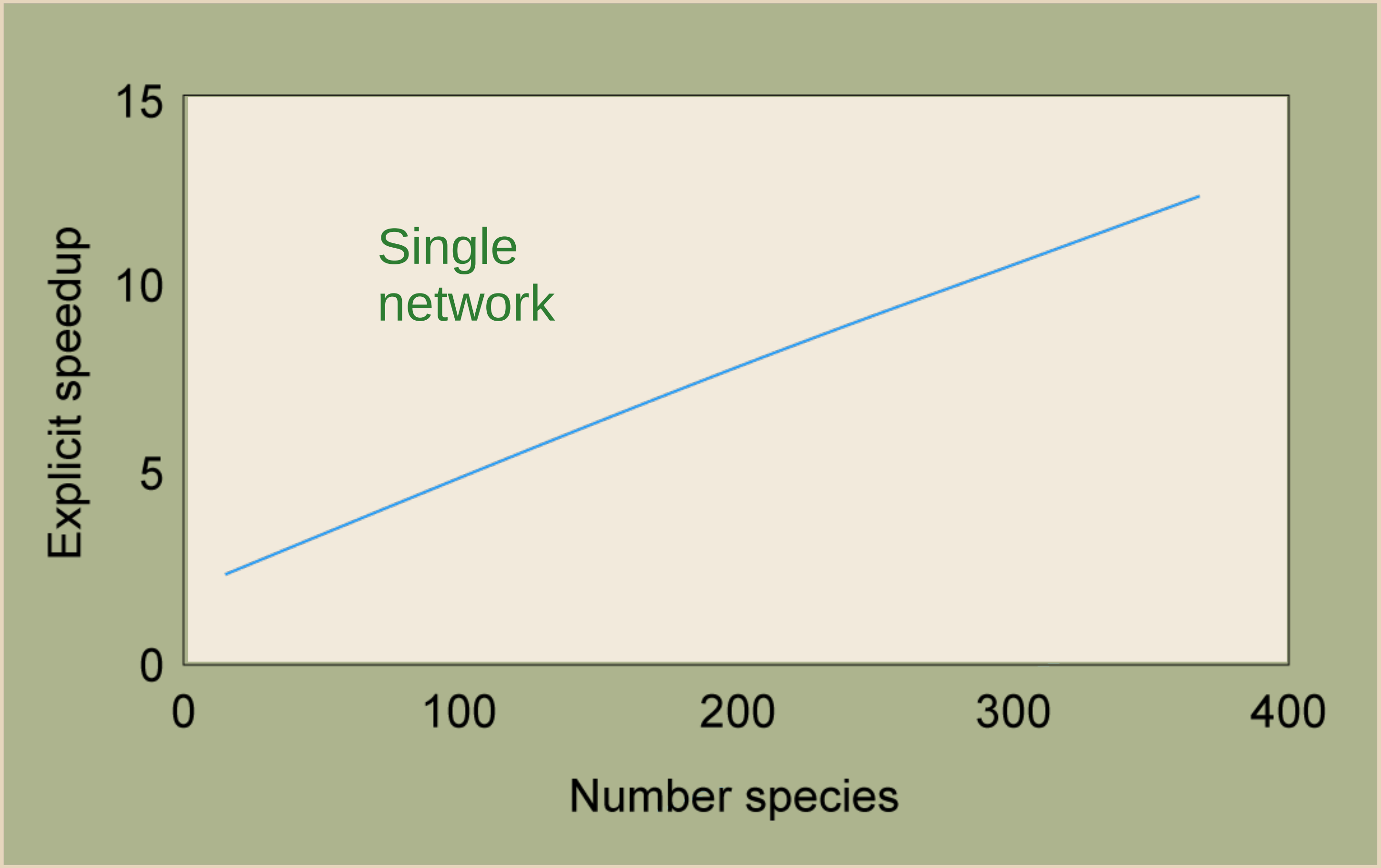
Example: Explicit Integration for a Nova Simulation





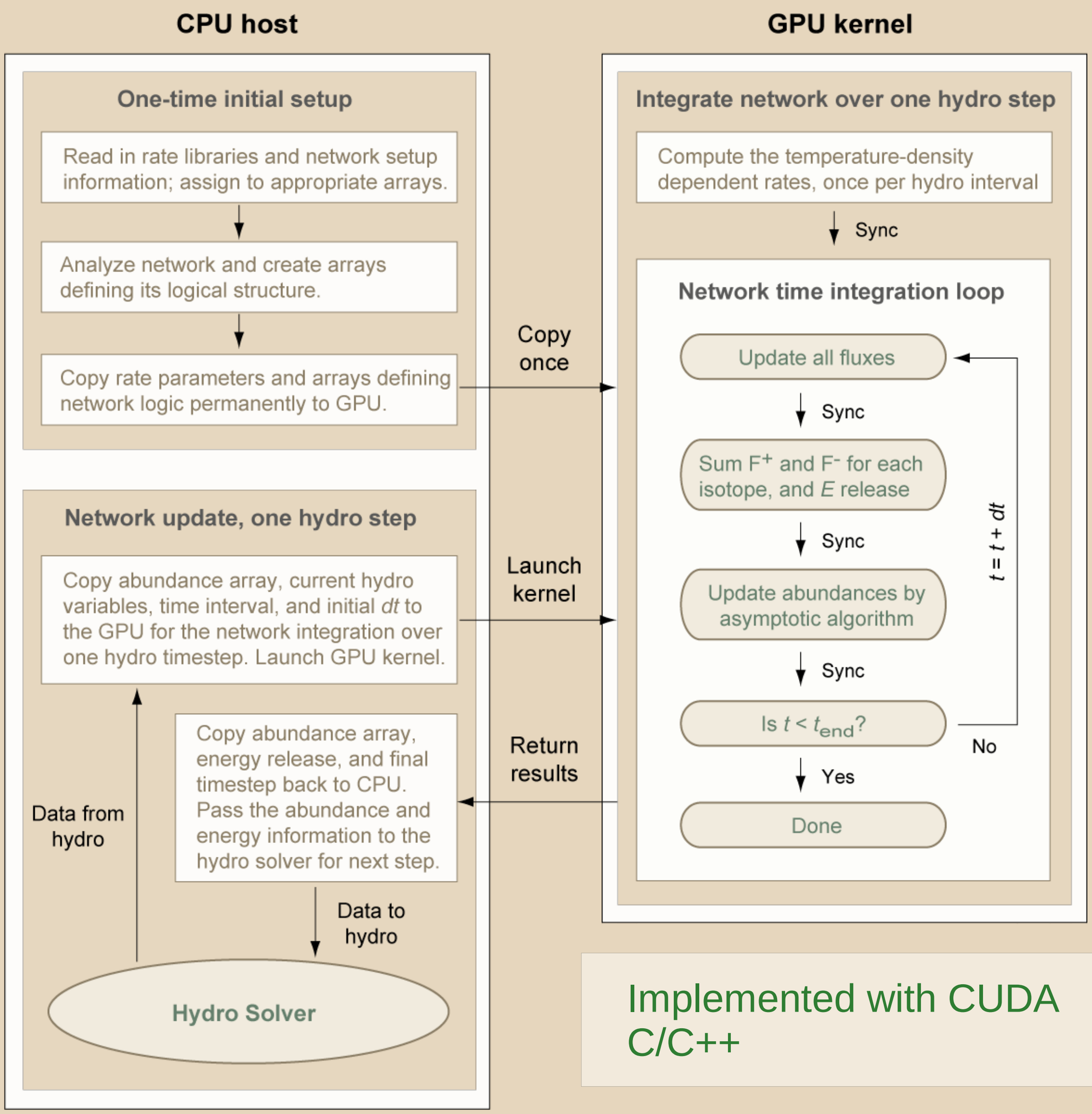
Method	Steps	Speed
Implicit	1332	1
Asy	935	10
QSS	777	12

Summary of Results: Explicit vs Implicit Speedup for a Single Network



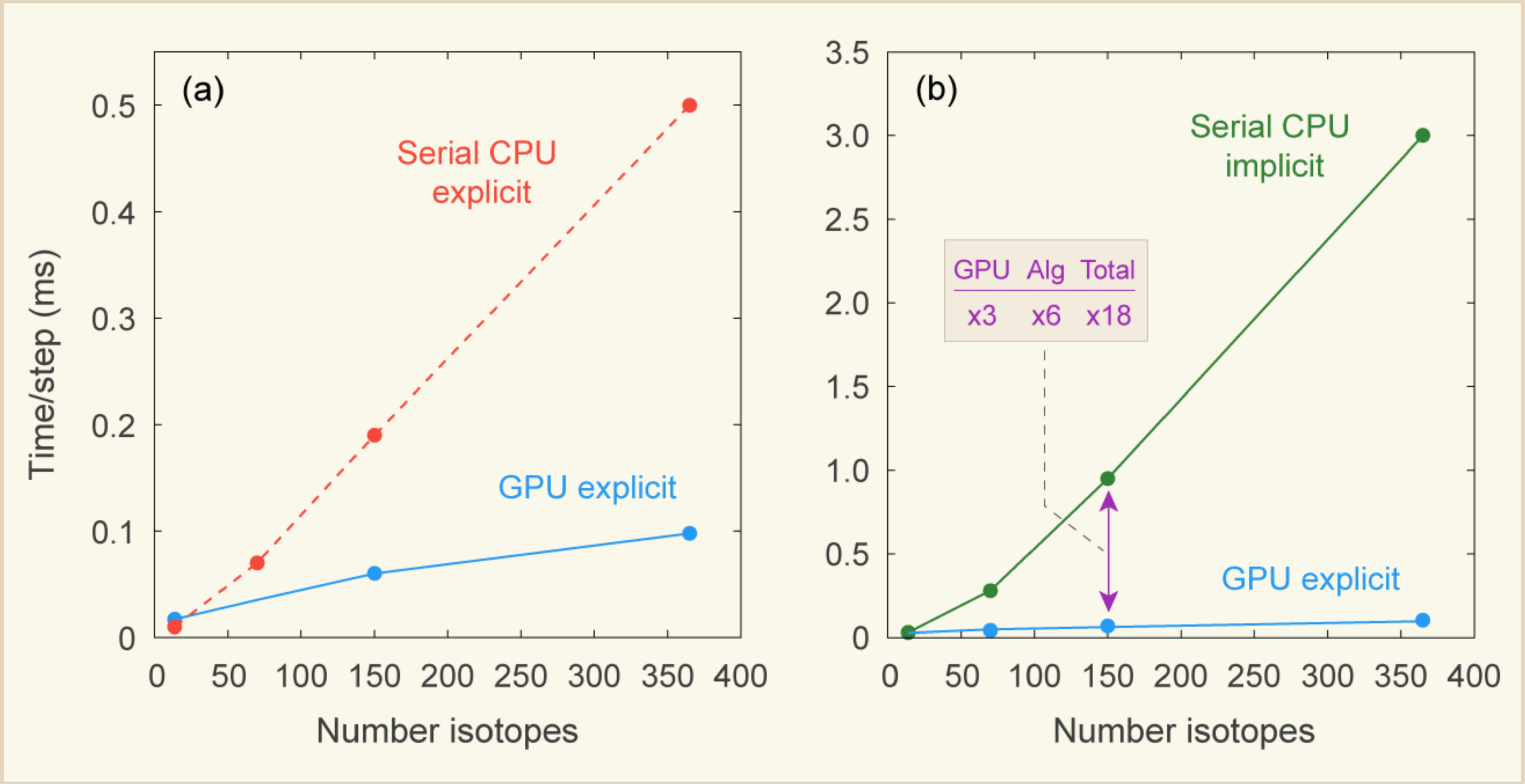
Thus our new algorithms can give a speed increase of about an order of magnitude for networks with several hundred species. Now let us consider the *role of modern hardware in this problem.*

GPU Acceleration for the Network

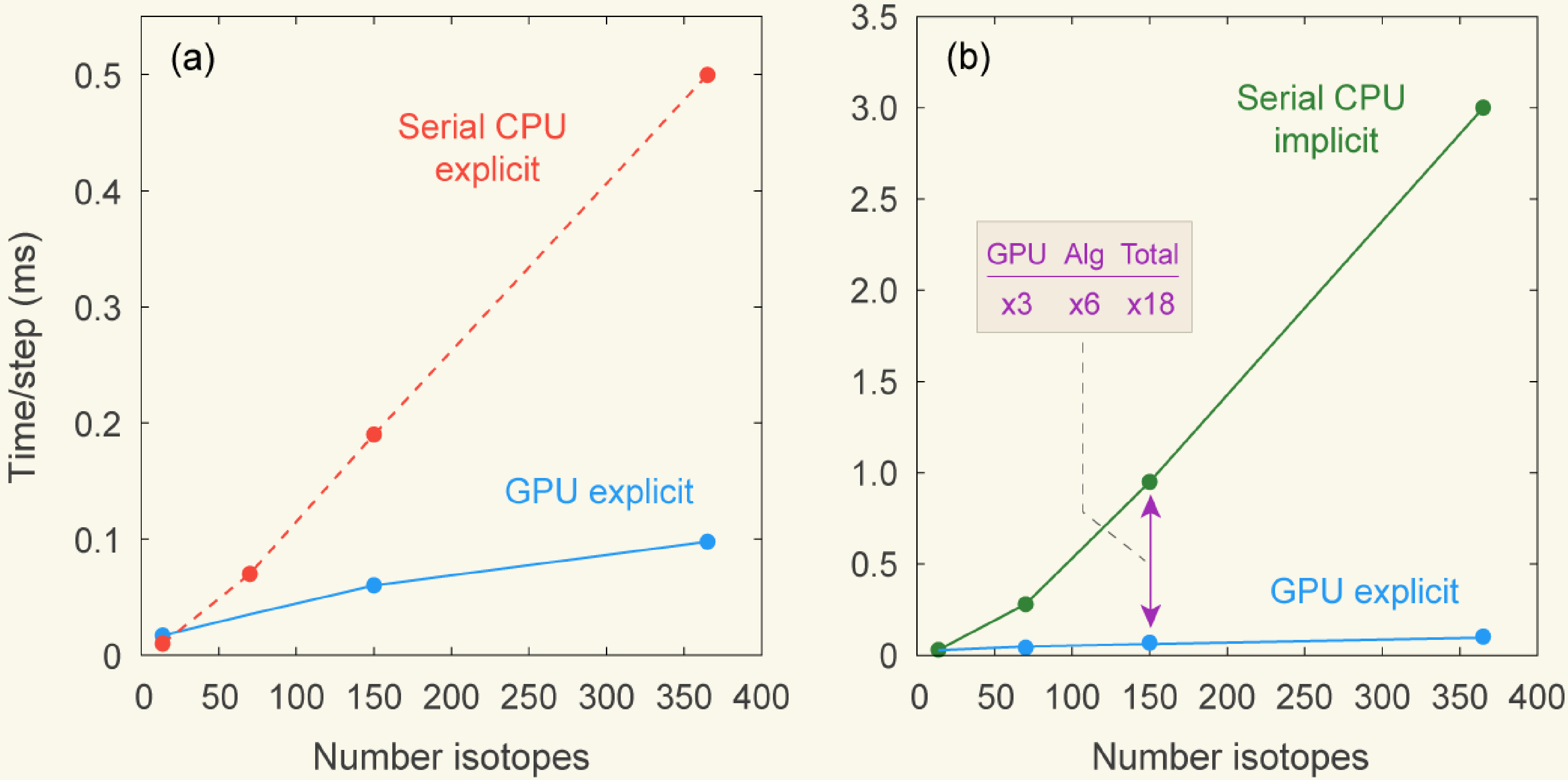


Implemented with CUDA C/C++

Scaling for Single Networks

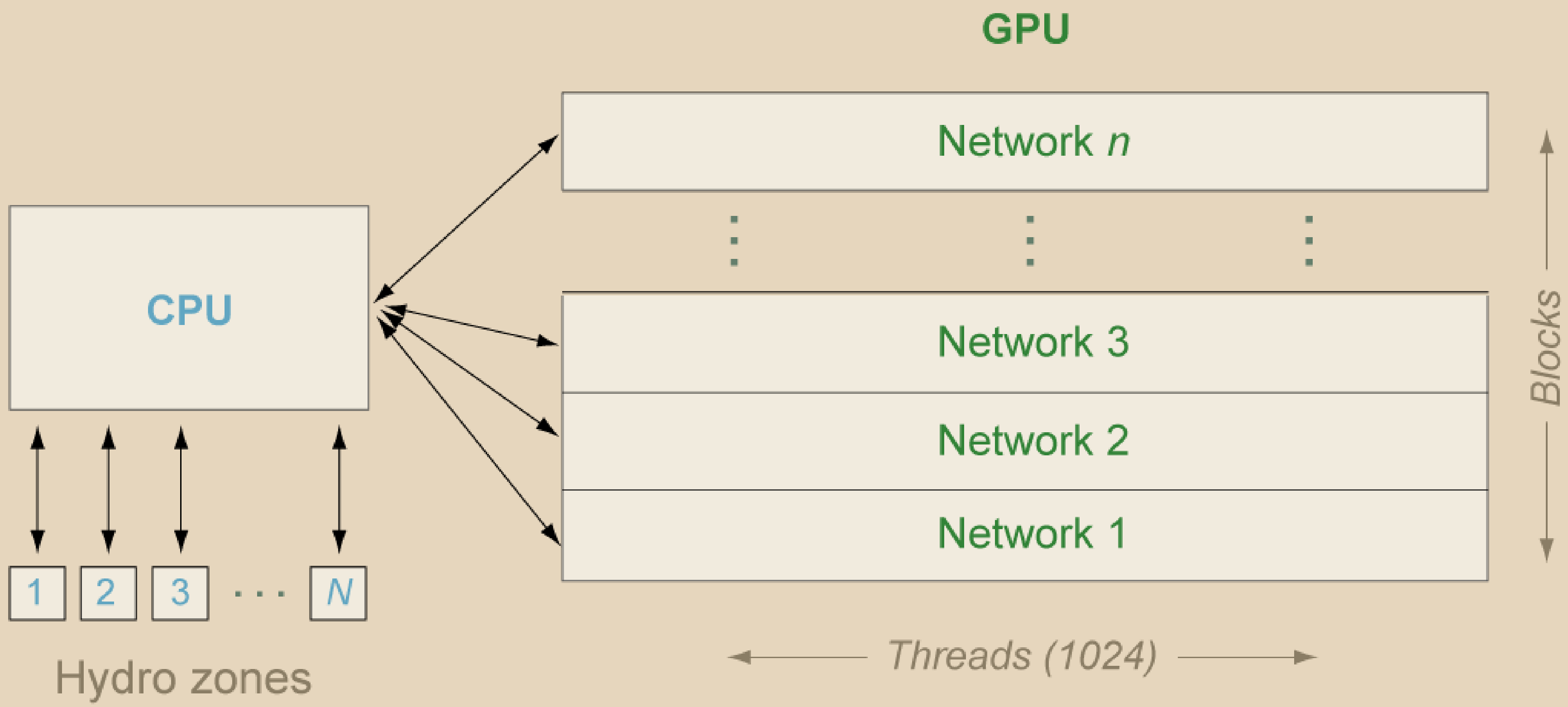


Scaling for a Single Network



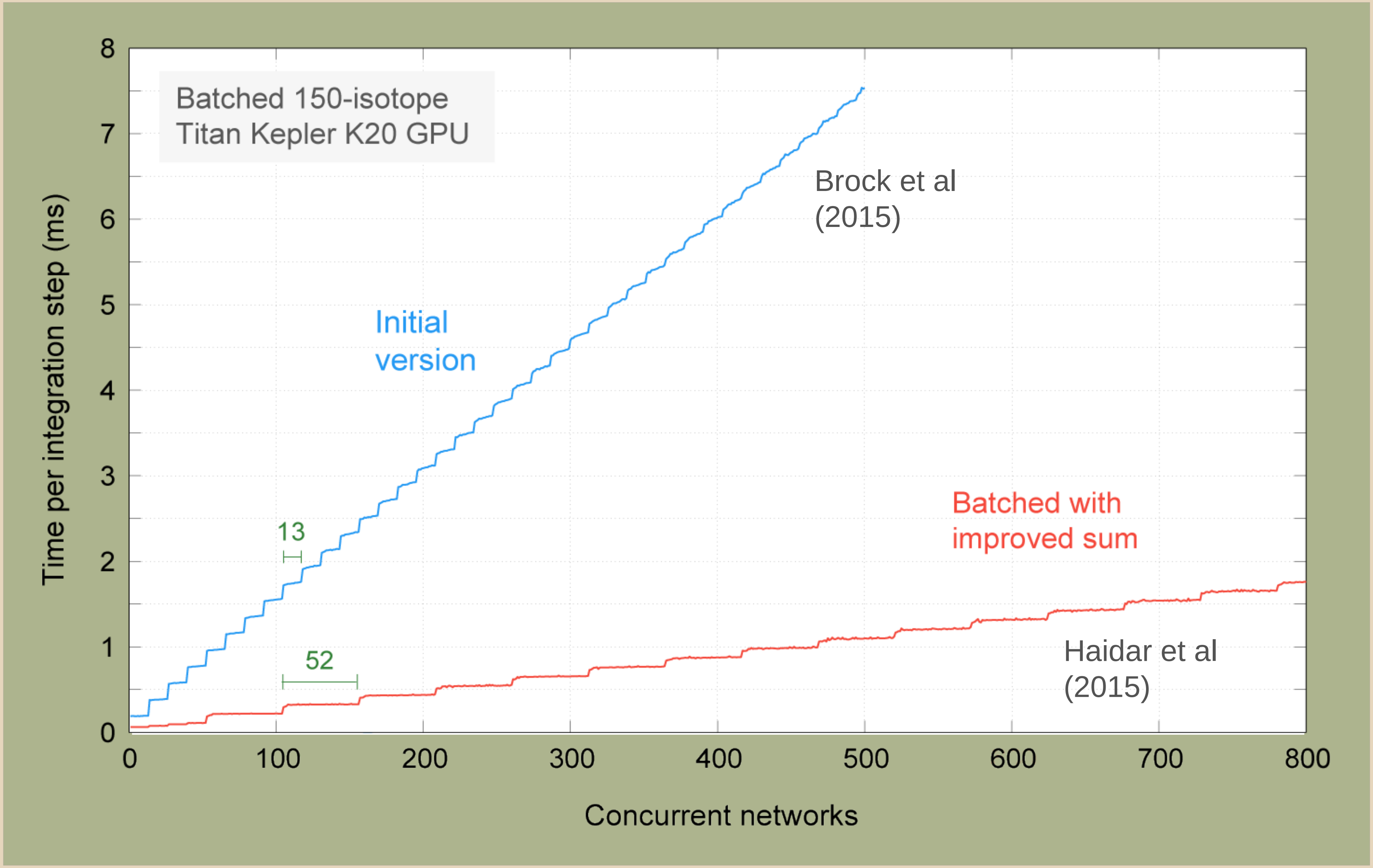
This is impressive speedup but *a single network utilizes only a small fraction of available GPU threads*. Greater efficiency requires that we *give the GPU more work.*

Stacking Multiple Networks on a GPU

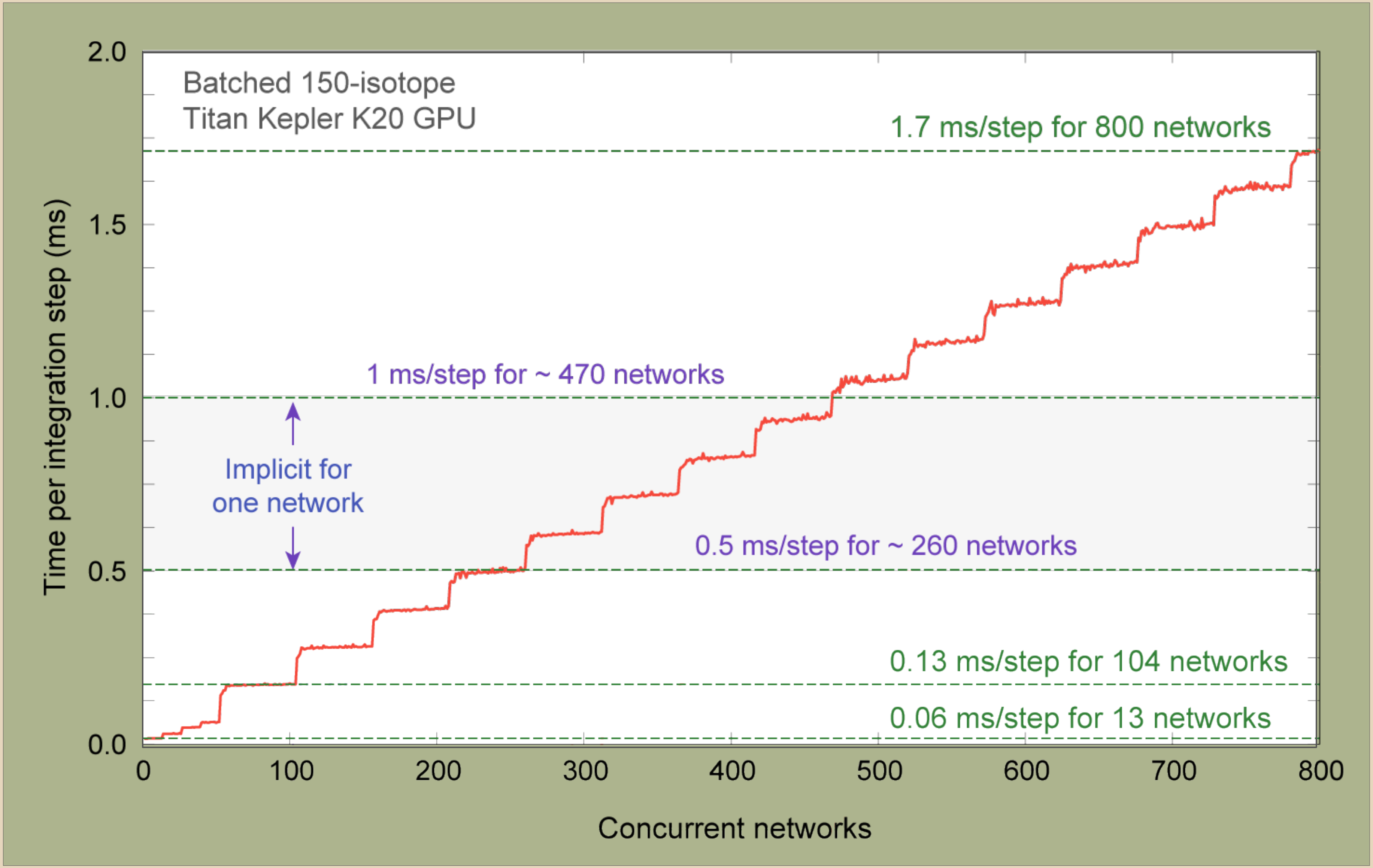


Thus, not only might it be possible to run one network of realistic size faster than is now feasible, it may be possible to run many such networks faster than it is now possible to run one such network.

Timing: Concurrent Network Launches



Timing: Concurrent Network Launches



Summary

- Our new algebraically-stabilized explicit algorithms are intrinsically faster than standard implicit algorithms by factors of 5-10 for networks with several hundred species.
- For a single network, GPU acceleration increases this to a factor of ~20-40 for networks with several hundred species.
- The GPU is capable of running ~250-500 networks in parallel in about the same length of time that a standard implicit code can run one such network (on a CPU or a GPU).
- These potential speed increases may mean that much more realistic kinetic networks coupled to hydrodynamics are now feasible in a range of large-scale problems in astrophysics and other disciplines.

Summary

- More generally, entire classes of stiff equations may be amenable to solution by these new methods. Some examples being considered:
  - Thermonuclear networks in astrophysics. Example: realistic Type Ia supernova simulations.
  - Atmospheric chemical networks in NOAA real-time continental weather simulation (urban-scale ozone forecast, etc)
  - Chemical and aerosol networks in climate modeling
  - Solution of neutrino transport equations in core-collapse supernovae
  - Fuel depletion in fission reactors
  - Simulation of reactor (fusion or fission) containment damage by radiation, and related materials science.

References

Algebraic Stabilization of Explicit Numerical Integration for Extremely Stiff Reaction Networks, Mike Guidry, J. Comp. Phys. 231, 5266-5288 (2012). [ArXiv:1112.4778]

Explicit Integration of Extremely-Stiff Reaction Networks: Quasi-Steady-State Methods, M. W. Guidry and J. A. Harris, Comput. Sci. Disc. 6, 015002 (2013) [ArXiv:1112.4750]

Explicit Integration of Extremely-Stiff Reaction Networks: Partial Equilibrium Methods, M. W. Guidry, J. J. Billings, and W. R. Hix, Comput. Sci. Disc. 6, 015003 (2013) [arXiv: 1112.4738]

Explicit Integration of Extremely-Stiff Reaction Networks: Asymptotic Methods, M. W. Guidry, R. Budiardja, E. Feger, J. J. Billings, W. R. Hix, O. E. B. Messer, K. J. Roche, E. McMahon, and M. He, Comput. Sci. Disc. 6, 015001 (2013) [ArXiv: 1112.4716]

Explicit Integration with GPU Acceleration for Large Kinetic Networks, Ben Brock, Andrew Belt, Jay Billings, and Mike Guidry, submitted to J. Comp. Phys. [arXiv:1409.5826]

arXiv = http://arxiv.org/

Collaborators

- Ben Brock, *University of Tennessee*
- Daniel Shyles, *University of Tennessee*
- Azzam Haidar, *University of Tennessee*
- Stan Tomov, *University of Tennessee*
- Jay Billings, *Oak Ridge National Laboratory*
- Andrew Belt, *University of Tennessee*
- Mike Guidry, *University of Tennessee*