3-1-2021

Additional Spatial Dimensions Can Help Speed Up Computations

Luc Longpre
The University of Texas at El Paso, longpre@utep.edu

Olga Kosheleva
The University of Texas at El Paso, olgak@utep.edu

Vladik Kreinovich
The University of Texas at El Paso, vladik@utep.edu

Comments:

Technical Report: UTEP-CS-21-30

Published in *Proceedings of the 14th International Workshop on Constraint Programming and Decision Making CoProd'2021*, Szeged, Hungary, September 12, 2021.

Additional Spatial Dimensions Can Help Speed Up Computations

Luc Longpré, Olga Kosheleva, and Vladik Kreinovich

Abstract While we currently only observe 3 spatial dimensions, according to modern physics, our space is actually at least 10-dimensional. In this paper, on different versions of the multi-D spatial models, we analyze how the existence of the additional spatial dimensions can help computations. It turns out that in all the versions, there is some speed up – moderate when the extra dimensions are actually compactified, and drastic if extra dimensions are separated by a potential barrier.

1 Computations and Space Dimensions: How They Are Related and What Are the Remaining Open Problems

Many computational problems require too much computation time. It is known that many practical computational problems are NP-hard; see, e.g., [4, 6]. This means, crudely speaking, that unless it turns out that P = NP (which most computer scientists do not believe to be possible), any algorithm that always solves the corresponding problem will require, at least for some inputs of reasonably large size, an unrealistically long time to solve – e.g., time larger than the lifetime of the Universe.

Parallelization can help – at least to some extent. If for a person, some task takes too much time, this person can (and does) ask for help. When two or more people

Luc Longpré
Department of Computer Science, University of Texas at El Paso, 500 W. University
El Paso, TX 79968, USA, e-mail: longpre@utep.edu

Olga Kosheleva
Department of Teacher Education, University of Texas at El Paso, 500 W. University
El Paso, TX 79968, USA, e-mail: olgak@utep.edu

Vladik Kreinovich
Department of Computer Science, University of Texas at El Paso, 500 W. University
El Paso, TX 79968, USA, e-mail: vladik@utep.edu

work on some task, they can perform it faster. Similarly, when a certain computational task requires too much time on a single computer, a natural way to speed up computations is to divide the original task between several computers – i.e., to parallelize computations. Many modern high-performance computers consists of thousands of processors working in parallel on the same task, and for many computational tasks, this indeed leads to a drastic speed-up.

Fundamental limitations of parallelization speed-up. In spite of the numerous successes of parallel computations, in general, parallelization is not a panacea: this idea has limitations. Some of these limitations are technical. These limitations will hopefully be overcome in the future. However, as we will show, there are also fundamental limitations on how much speed-up can be achieved by parallelization; see, e.g., [5].

Indeed, let us assume that we have a parallel computer that finishes its computations in time T_{par}. Let us show how we can simulate its computations sequentially. According to modern physics (see, e.g., [1, 8]), the speed of all processes is bounded by the speed of light c. During the time T_{par}, the information from the processors must reach the user. This means that the processors that participate in this computation must be located within the distance $R \stackrel{\text{def}}{=} c \cdot T_{\text{par}}$, i.e., in geometric terms, inside the sphere of radius R centered at the user location.

The overall volume of this area is equal to

$$V = \frac{4}{3} \cdot \pi \cdot R^3 = \frac{4}{3} \cdot \pi \cdot c^3 \cdot T_{\text{par}}^3.$$

Thus, if we denote by ΔV the smallest possible volume of a single processor, then the number of processor N_{proc} that can fit inside this sphere cannot exceed the value

$$N_{\text{proc}} \le N_{\max} \stackrel{\text{def}}{=} \frac{V}{\Delta V} = \frac{4}{3 \cdot \Delta V} \cdot \pi \cdot c^3 \cdot T_{\text{par}}^3. \tag{1}$$

Whatever we can compute in parallel on N_{proc} processors, we can also compute sequentially, if we first simulate all the first steps of all the processor, then all the second steps of all the processors, etc. This way, each step of the parallel computer requires N_{proc} steps of the sequential computer. Thus, what was computed on a parallel computer in time T_{par} can be computed on a sequential computer in time $T_{\text{seq}} = N_{\text{proc}} \cdot T_{\text{par}}$.

Due to the inequality (1), we have

$$T_{\text{seq}} \le \frac{4}{3 \cdot \Delta V} \cdot \pi \cdot c^3 \cdot T_{\text{par}}^3 \cdot T_{\text{par}} = C \cdot T_{\text{par}}^4, \tag{2}$$

where we denoted

$$C \stackrel{\text{def}}{=} \frac{4}{3 \cdot \Delta V} \cdot \pi \cdot c^3.$$

So, if the fastest time that it takes for a sequential computer to solve a problem is T, the fastest time T_{par} that this same problem can be solved on a parallel computer is bounded by the inequality $T \le T_{\text{seq}} \le C \cdot T_{\text{par}}^4$, thus

$$T_{\text{par}} \ge C^{-1/4} \cdot T^{1/4}. \qquad (3)$$

This implies that by using parallelization, we can speed up, at best, to the 4-th root of the sequential time. This is good, but not ideal: if the original sequential time T was exponential – as for NP-hard problems – the parallel time is still exponential.

Extra dimensions: a brief reminder. The above argument assumes that we live in s 3-dimensional space. However, according to modern physics, the requirement that quantum field theory is consistent implies that the dimension of space is at least 10; see, e.g., [2, 7, 8].

Resulting challenge and what we do in this paper. A natural question is: how does the presence of these extra spatial dimensions affect computations?

This is the question that we study in this paper.

2 First (Naive) Idea and Why It Is Naive

A seemingly natural idea. At first glance, the situation is straightforward: if instead of 3 spatial dimensions we have $d > 3$ dimensions, then the volume of the area inside the sphere of radius R is equal to $V = c_d \cdot R^d$ for some constant c_d. Taking into account that $R = c \cdot T_{\text{par}}$, we conclude that $V = c_d \cdot c^d \cdot T_{\text{par}}^d$. Thus, the number N_{proc} of processors is bounded by the number

$$N_{\text{proc}} \le N_{\max} \overset{\text{def}}{=} \frac{V}{\Delta V} = \frac{c_d}{\Delta V} \cdot c^d \cdot T_{\text{par}}^d.$$

Hence, this parallel computation can be simulated on a sequential computer in time

$$T_{\text{seq}} \le N_{\text{proc}} \cdot T_{\text{par}} = \frac{c_d}{\Delta V} \cdot c^d \cdot T_{\text{par}}^d \cdot T_{\text{par}} = C_d \cdot T_{\text{par}}^{d+1},$$

where this time

$$C_d \overset{\text{def}}{=} \frac{c_d}{\Delta V} \cdot c^d.$$

So, instead of the previous rather-high lower bound $T_{\text{par}} \ge \text{const} \cdot T_{\text{seq}}^{1/4}$, we get a much better lower bound $T_{\text{par}} \ge \text{const} \cdot T_{\text{seq}}^{1/(d+1)}$, with $d \ge 10$.

Why this idea is naive. The above result looks good, but it is based on the simplified idea that extra spatial dimensions are similar to the current three ones. In reality, the fact that we currently observe only three dimensions means that different spatial dimensions are different.

There are two possible approaches to how to explain that other dimensions are not yet observable. In this section, we describe these two approaches, and in the following sections, we analyze how these approaches affect computations.

First approach: actual compactification. The first natural approach is to conclude that since we cannot observe any changes in other spatial dimensions, this means that these dimensions are very small in size – e.g., that each of these dimensions represents not a line, but a circle of a small radius.

Second approach. The second natural approach is to assume that while all our processes are happening in a very small fragment of the additional dimensions. these dimensions actually have larger size – only due to some physical reasons, we cannot leave this small fragment. An analogy is when we are in a narrow valley between two mountain ranges: in principle, we can get out of this valley, but this requires climbing high mountains – and for that, we will need lots of energy and probably special equipment, which few of us have.

What we will do now. Let us see how both physically realistic versions of extra spatial dimensions can affect computations.

3 First Approach: How Actual Compactification Affects Computations

It all boils down to computing the volume. The above arguments about the limits to parallelization were based on computing the volume V of the inside of the sphere of radius $R = c \cdot T_{\text{par}}$, where c is the speed of light and T_{par} is the computation time. In the analysis of the 3-D situation, we used the formula for the volume of a sphere in the 3-D space. To see how the resulting calculations will change in the multi-D space, we need to find, for this space, what is the corresponding volume V.

Let us compute this volume. To find this volume, let us recall that the distance between the two points $x = (x_1, x_2, \ldots)$ and $y = (y_1, y_2, \ldots)$ in the multi-D space is equal to

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2 + (x_4 - y_4)^2 + \ldots}.$$

For reasonable computation time T_{par}, the radius $R = c \cdot T_{\text{par}}$ is large, and thus, is much larger than the size s_e of each extra dimension: remember that this size is so small that we do not notice these extra spatial dimensions. So, the terms

$$(x_4 - y_4)^2, \ldots$$

corresponding to differences in extra dimensions – and which are of order s_e^2 – are much much smaller than the terms $(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2$ describing the distance in the 3-D space. Hence, with high accuracy, we can safely assume that the distance between the two multi-D points is equal to the distance between their 3-D parts:

$$d(x,y) \approx d_3(x,y) \stackrel{\text{def}}{=} \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}.$$

Therefore, the set of all the points which are at distance $\le R$ from the user can be described as follows: we take all the points (x_1, x_2, x_3) from the corresponding 3-D sphere, and for each of these points, we consider all possible combinations $(x_4, \ldots)$ of additional spatial coordinates.

The size of each additional coordinate is s_e, and in a d-dimensional space, there are $d-3$ additional spatial coordinates. Thus, the overall volume of the additional part of s_e^{d-3}, and the overall volume of the sphere in d-dimensional space is equal to $\frac{4}{3} \cdot \pi \cdot R^3 \cdot s_e^{d-3}$.

How many processors can we fit now? The multi-D volume ΔV of a processor can be obtained by multiplying its 3-D volume ΔV_3 by its volume ΔV_e in the extra dimensions. If the size of the processor in additional dimensions is s_e, then we get the exact same number of processors as in the 3-D case, no gain at all from the existence of additional spatial dimensions.

However, if we manage to decrease the size of a processor in extra dimensions to less than s_e, so that the volume ΔV_e of a processor in the extra dimensions is smaller than s_e^{d-3}, then, by dividing the overall multi-D volume by the volume of a single processor, we get the new value for the number of processors:

$$N_{\text{proc}} \le N_{\max} = \frac{V}{\Delta V} = \frac{\frac{4}{3} \cdot \pi \cdot R^3 \cdot s_e^{d-3}}{\Delta V_3 \cdot \Delta V_e} = \frac{4}{3 \cdot \Delta V_3} \cdot \pi \cdot R^3 \cdot \frac{s_e^{d-3}}{\Delta V_e}.$$

Since we consider the case when $\Delta V_e < s_e^{d-3}$, this number of processors is larger than the corresponding 3-D number of processors

$$\frac{4}{3 \cdot \Delta V_3} \cdot \pi \cdot R^3$$

by a factor of

$$C = \frac{s_e^{d-3}}{\Delta V_e} > 1.$$

Conclusion for this approach. In the first approach to multi-D space-time – when all extra dimensions are actually compactified – after an appropriate level of miniaturization, we will be able to get a C times increase in number of processors that we can fit into each area – and thus, in principle, a constant times computation speed-up.

Comment. This is not as spectacular as we could imagine based on the naive approach, but any speed up is good.

4 Second Approach: How It Affects Computations

At first glance. If we limit ourselves to the same small area of extra dimensions in which all observable processes take place, then we get the exact same situation as in the first approach – and thus, we can get the same constant times increase, where the constant depends on how successful we are in minituarizing our processors.

But now we have another option. However, in the second approach, we do not have to limit ourselves to the small area that contains all observable processes: there are other areas as well, it is just that these areas are difficult to reach: since going there requires a lot of energy, thus preventing usual particles from going there.

What if we apply this considerable amount of energy and reach these additional areas? What do we gain with respect to computations?

First gain: all the promises of the naive approach turn out to be true. If we are allowed to use a significant area in extra dimensions, then we can have all the advantages promised by the above-described naive approach: namely, instead of being able to fit $\sim T_{\rm par}^3$ processors into an area of radius $R = c \cdot T_{\rm par}$, we can fit a much larger amount of $\sim T_{\rm par}^d$ processors. Thus, instead of the possibility to reduce the sequential computation time $T_{\rm seq}$ to $T_{\rm par} \sim T_{\rm seq}^{1/4}$, we can get a much more drastic speed-up $T_{\rm par} \sim T_{\rm seq}^{1/(d+1)}$.

Interestingly, there is an additional speed-up. The fact that *all* the processes are limited to a narrow area of values of extra spatial dimensions means, in effect, that this limitation is the property of the underlying space-time, not of any specific physical field. In other words, this means that the space-time is not as flat as the space-time of our usual 3D space – that would have enabled particles to easily go in all possible spatial directions – but rather curved.

According to General Relativity theory – the theory that describes curved space-time in modern physics – in a curved space-time, free particles move along geodesic lines, i.e., lines in which the resulting proper time Δs between the each two locations is the shortest possible. In terms of coordinate time t, this overall proper time can be computed by adding up proper times $ds = \dfrac{ds}{dt} \cdot dt$ corresponding to different parts of the trajectory, i.e., as $\Delta s = \displaystyle\int \frac{ds}{dt}\, dt$. According to General Relativity, the ratio $\dfrac{ds}{dt}$ is, in general, smaller than 1: in a gravitational field, all the processes slow down, and if this field is very strong – e.g., near a black hole – then it can slow down drastically: when the outside world measures 10 years, people near the black hole will only count several months.

In [3], we considered possible computational consequences of this effect in the 3D space. Interestingly, in the second approach to the multi-D cases, we have an additional possibility to use this effect. Namely, the fact that for all the particles, the optimal path is by going via the narrow zone of observable processes means that in this zone, the ratio $\dfrac{ds}{dt}$ is much smaller than in the neighboring zones – just like the fact that the fastest way to get from two points in the US usually involves taking a freeway is an indication that the allowed speed on the freeway is larger than on all other roads.

For example, if we are in the vicinity of a gravitating body, where the ratio $\frac{ds}{dt}$ is smaller than 1 – and which is thus an analogue of a freeway – particles will tend to move close to this vicinity, which we observe as gravitational attraction. The stronger the gravitational field, the smaller the ratio $\frac{ds}{dt}$ and thus, the more probable it is that the particles will bend towards this vicinity – so the larger the observed gravitational attraction.

In our multi-D case, the fact that in the neighborhood of our zone the value of the ratio is much larger than in the zone itself means that during the same time Δt, the proper time Δs in this neighborhood will be larger than in our zone. In other words, during the same coordinate time, the processor located in the neighborhood will be able to perform more operations than a processor that stays in our zone. Thus, we will get an additional speed-up.

Conclusion for this approach. In the second approach,

- not only we can have more processor working in parallel – by placing additional processors outside the narrow zone where the observable processes occur,
- but also the processors placed outside this zone will compute much faster than the ones in the zone, which will lead to an additional speedup.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), and HRD-1834620 and HRD-2034030 (CAHSI Includes).

It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478.

References

1. R. Feynman, R. Leighton, and M. Sands, *The Feynman Lectures on Physics*, Addison Wesley, Boston, Massachusetts, 2005.
2. M. B. Green, J. H. Schwarz, and E. Witten, *Superstring Theory*, Vols. 1, 2, Cambridge University Press, 1988.
3. O. Kosheleva and V. Kreinovich, "Relativistic effects can be used to achieve a universal square-root (or even faster) computation speedup", In: A. Blass, P. Cegielsky, N. Dershowitz, M. Droste, and B. Finkbeiner (eds.), *Fields of Logic and Computation III*, Springer, 2020, pp. 179–189.
4. V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer, Dordrecht, 1998.

5. D. Morgenstein and V. Kreinovich, "Which algorithms are feasible and which are not depends on the geometry of space-time", *Geombinatorics*, 1995, Vol. 4, No. 3, pp. 80–97.
6. C. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, Massachusetts, 1994.
7. J. Polchinski, *String Theory*, Vols. 1, 2, Cambridge University Press, 1998.
8. K. S. Thorne and R. D. Blandford, *Modern Classical Physics: Optics, Fluids, Plasmas, Elasticity, Relativity, and Statistical Physics*, Princeton University Press, Princeton, New Jersey, 2017.