Journal of Uncertain Systems
Vol.6, No.x, pp.xx-xx, 2012
Online at: www.jus.org.uk

Computation in Quantum Space-Time Could Lead to a Super-Polynomial Speedup

Michael Zakharevich[1] and Vladik Kreinovich[2*]
[1] *Department of Mathematics, Stanford University*
Stanford, CA 94305, USA, ymzakharevich@yahoo.com
[2] *Department of Computer Science, University of Texas at El Paso*
El Paso, Texas 79968, USA, olgak@utep.edu

Received 1 June 2011; Revised 14 July 2011

Abstract

In theoretical computer science, researchers usually distinguish between feasible problems (that can be solved in polynomial time) and problems that require more computation time. A natural question is: can we use new physical processes, processes that have not been used in modern computers, to make computations drastically faster – e.g., to make intractable problems feasible? Such a possibility would occur if a physical process provides a super-polynomial (= faster than polynomial) speed-up.

In this direction, the most active research is undertaken in quantum computing. It is well known that quantum processes can drastically speed up computations; however, there are proven super-polynomial quantum speedups of the overall computation time.

Parallelization is another potential source of speedup. In Euclidean space, parallelization only leads to a polynomial speedup. We show that in quantum space-time, parallelization could potentially lead to (provably) super-polynomial speedup of computations.

Keywords: feasible computations, quantum computing, parallelization, quantum space-time, super-polynomial speedup

1 Which Problems Are Feasible: Brief Reminder

In theoretical computer science (see, e.g., [8]), researchers usually distinguish between:

- problems that can be solved in polynomial time (i.e., in time that is bounded by a polynomial $P(n)$ of the length n of the input) and
- problems that require more computation time.

Problems solvable in polynomial time are usually called *feasible*, while others are called *intractable*.

Of course, this definition is not a perfect description of the intuitive notion of feasibility. For example, an algorithm that requires $10^{100} \cdot n$ steps is polynomial time but not practically feasible. However, this is the best available definition of feasibility.

2 Is Speedup Possible?

The big challenge is that some computational problems are intractable – i.e., require algorithms which are too slow. At first glance, if a problem is intractable in this sense, there is not much we can do about it.

However, there is hope. Traditional notions of computational complexity are based on the assumption that we only use the physical processes which are traditionally used in computers. So, a natural question is: what if we use new physical processes, processes that have not been used in modern computers? Will the use of these physical processes enable us to make computations drastically faster? Will these new physical processes make intractable problems feasible?

Such a possibility would occur if a physical process provides a super-polynomial (= faster than polynomial) speedup.

*Corresponding author. Email: vladik@utep.edu (V. Kreinovich).

3 Quantum Computing: A Possible Path to a Super-Polynomial Speedup

In this direction, the most active research is undertaken in quantum computing. It is well known that quantum processes can speed up computations; see, e.g., [7].

For example:

- in non-quantum computing, a search in an un-sorted list of size N requires, in the worst case, at least N computational steps,
- in contrast, in quantum computing, Grover's algorithm enables us to search in an un-sorted list of size N in time $\sqrt{N}$.

Grover's algorithm can be applied to problems for which, currently, no algorithm is known which is drastically faster than exhaustive search. A good example of such a problem is the well-known propositional satisfiability problem SAT:

- *Given:* a propositional formula $F(x_1, \ldots, x_n)$, i.e., an expression that is formed from variables $x_1, \ldots, x_n$ by applying the propositional connectives & ("and"), $\vee$ ("or"), and $\neg$ ("not").
- *Find:* values $x_1, \ldots, x_n \in \{\text{false}, \text{true}\} = \{0, 1\}$ that make the given formula $F(x_1, \ldots, x_n)$ true – or, if no such values exist, return the corresponding message.

In principle, this problem can be solved by exhaustive search, i.e., by trying all 2^n possible combinations of truth values $x_i = 1$ ("true") and $x_i = 0$ ("false"). This exhaustive search requires 2^n computational steps. By using Grover's algorithm, we can reduce the computation time from 2^n to $\sqrt{2^n} = 2^{n/2}$.

This is a drastic speedup, but this speedup is still polynomial, it does not allow us to replace the original longer-than-polynomial time with a polynomial one. Specifically, in this case, if we denote by $T_c(n)$ the non-quantum computation time, then the quantum computation time is $T_q(n) = \sqrt{T_c(n)}$. So, if the quantum computation time $T_q(n)$ is polynomial, so is the non-quantum computation time $T_c(n) = T_q^2(n)$.

Some known quantum algorithms are exponentially faster than the best known non-quantum ones. A known example of such a situation is Shor's algorithm for factoring large integers. However, it is still not clear whether a similar fast non-quantum algorithm is possible.

In general, there are no *proven* super-polynomial quantum speedups of the overall computation time.

4 Parallelization – Another Potential Source of Speedup

Parallelization is another potential source of speedup: if we have several computers working in parallel, then we can perform computations faster.

Parallelization in Euclidean space. In the usual (Euclidean) space, parallelization only leads to a polynomial speed-up; see, e.g., [3, 4, 6]. The main reason for this limitation is that, according to modern physics, the speed of all the physical processes is bounded by the speed of light c. Thus, in time T, we can only reach computational units at a distance not exceeding $R = c \cdot T$. In other words, within time T, we can only use computational devices located inside the sphere of radius $R = c \cdot T$.

In the Euclidean space, the volume $V(R)$ of this area (inside of the sphere of radius $R = c \cdot T$) is equal to

$$V(R) = \frac{4}{3} \cdot \pi \cdot R^3 = \frac{4}{3} \cdot \pi \cdot c^3 \cdot T^3,$$

and is, thus, proportional to T^3. So, if we denote by ΔV the volume of the smallest possible computational device, we can conclude that for computations that last time T, we can use no more than $\leq \dfrac{V(R)}{\Delta V} \sim T^3$ computational elements.

If we denote by Δt, the smallest time of a single computational operation, we can conclude that during time T, each computational element can perform $\leq \dfrac{T}{\Delta t} \sim T$ computational steps. Thus, all $\sim T^3$ computational

elements can perform $\leq T^3 \cdot T = T^4$ computational steps. So, if we simulate this parallel computer element-by-element on a sequential computer, we will thus be able to perform the same computations in time $\sim T^4$.

Thus, if a parallel computer can perform computations in polynomial time $T \leq P(n)$, the same computations can be performed on a sequential computer in time $\leq C \cdot T^4 \leq C \cdot (P(n))^4$ which is still polynomial. In other words, parallelization in Euclidean space can only lead to a polynomial speedup.

Parallelization in non-Euclidean space. It is well known (see, e.g., [5]) that the actual space is not Euclidean, it is curved. Interestingly, in some non-Euclidean spaces – e.g., in Lobachevsky space, historically the first non-Euclidean space model – volume $V(R)$ grows exponentially with the radius:

$$V(R) \sim \exp(R) \gg \text{Polynomial}(R).$$

Thus, in principle, we can fit exponentially many processors within a sphere of radius $R = c \cdot T$ – and hence get a drastic (exponential) speed-up [4, 6].

The problem is that while there are physically reasonable space models that allow such an exponential speedup, these models are very hypothetic. The space models corresponding to mainstream cosmological models do not have this exponential growth and thus, only allow polynomial parallelization speedup.

5 Quantum Space-Time Models: a New Possibility of Speedup

Main idea. So far, we have described two separate approaches to computation speedup:

- use of quantum effects, and
- use of curved space-time.

In physics, the corresponding quantum and space-time effects are related. Specifically, non-quantum space-time is only an approximate description. A more accurate description of space-time requires that we take into account quantum effects, i.e., that we consider *quantum space-time models.*

Thus, it is reasonable to combine the two above approaches to computation speedup, and consider the use of quantum space-time models for computations. To do that, let us recall how quantum effects affect space-time; for details, see, e.g., [2, 5].

How quantum effects affect space-time: qualitative description. The main reason why quantum effects affect space-time is *Heisenberg's Uncertainty Principle.* Its most well-known form is that we cannot simultaneously measure the coordinate x and the momentum p with absolute accuracy:

- the accuracy Δx with which we can determine the coordinate x and
- the accuracy Δp with which we can determine the momentum p

are related by the inequality $\Delta p \cdot \Delta x \geq \hbar$, where $\hbar$ is Planck's constant, one of the fundamental constants behind quantum physics.

A similar (less well-known) inequality holds for energy E and time t:

- the accuracy ΔE with which we can determine the energy E and
- the accuracy Δt with which we can determine the time t

are related by the inequality $\Delta E \cdot \Delta t \geq \hbar$. As a result, when we restrict ourselves to a region of spatial size ε, i.e., to times $\Delta t \approx \varepsilon/c$, we get the energy uncertainty $\Delta E \sim \hbar \cdot \varepsilon^{-1}$.

When $\varepsilon \to 0$, we get $\Delta E \to \infty$. So, when size ε of the region is small, a lot of energy enters this region. According to the famous relativity theory formula $E = m \cdot c^2$, this energy has the mass $m = E \cdot c^{-2}$, and this mass causes a gravitational field. According to General Relativity theory, the gravitational field means curving space-time, so we can say that the energy ΔE *curves* the space-time.

The smaller the scale ε, the more energy we have, so the more the space-time is curved. Hence, on a small scale, space-time is very curved ("foam"-like) [5].

How quantum effects affect space-time: quantitative description. The above qualitative description can be transformed into a quantitative one [2].

Indeed, as we have mentioned, the total energy of all the fluctuations in area of size ε is equal to $E \sim \hbar \cdot \varepsilon^{-1}$. To find out how many fluctuations of smaller size $k \cdot \varepsilon$ ($k < 1$, $k \approx 1$) can fit into this area, let us estimate the energy δE of a single fluctuation of this smaller size.

For that, we need to take into account that in modern physics, all fundamental physical equations are determined by minimizing *action*, crudely speaking, a product of energy and time; see, e.g., [1]. In Einstein's General Relativity, action is proportional to $L = \int R\, dV\, dt$, where R is a scalar curvature [1, 5]. Since action is energy times time, we have $\delta E \cdot \Delta t \sim \int R\, dV \approx R \cdot V \cdot \Delta t$, hence, by dividing both sides by δt, we get $\delta E \sim R \cdot V$. For a fluctuation of linear size $k \cdot \varepsilon \sim \varepsilon$, volume is $V \sim \varepsilon^3$ and curvature – which is inverse proportional to the square of the linear size – is $R \sim \varepsilon^{-2}$. Thus, we conclude that $\delta E \sim \varepsilon$.

So, the total number n of fluctuations of size $k \cdot \varepsilon$ in an area of size ε can be determined by dividing the total energy E of such fluctuations by the energy δE of a single fluctuation of size $k \cdot \varepsilon$:

$$n = \frac{E}{\delta E} \sim \hbar \cdot \varepsilon^{-2}.$$

Comment. The above physical derivations are described, in detail, in [2].

Application to speedup: analysis. What we are interested in is how many processors $N_q(\varepsilon)$ of size ε can we fit in a given region?

Ideally, we may be able to fit one processor (once it is sufficiently small) in each fluctuation. In this case, the desired number of processors is equal to the number $N_q(\varepsilon)$ of regions of size ε that can fit into a given region.

In every region of spatial size ε, we can fit $n \sim \hbar \cdot \varepsilon^{-2}$ regions of size $k \cdot \varepsilon$. Thus, the total number $N_q(k \cdot \varepsilon)$ of regions of size $k \cdot \varepsilon$ is equal to the total number $N_q(\varepsilon)$ of regions of size ε times $n \sim \hbar \cdot \varepsilon^{-2}$:

$$N_q(c \cdot \varepsilon) \approx \hbar \cdot \varepsilon^{-2} \cdot N_q(\varepsilon).$$

Let us use this functional equation to find $N(\varepsilon)$. By substituting $\varepsilon = 1$, $\varepsilon = k$, $\varepsilon = k^2$, etc., into the above formula, and taking into account that $N_q(1) \approx 1$, we conclude that:

- $N_q(k) \approx \hbar \cdot k^{-2}$;
- $N_q(k^2) \approx \hbar \cdot k^{-4} \cdot N_q(k)$, hence $N_q(k^2) \approx \hbar^2 \cdot k^{-(2+4)}$;
- $N_q(k^3) \approx \hbar \cdot k^{-6} \cdot N_q(k^2)$, hence $N_q(k^3) \approx \hbar^3 \cdot k^{-(2+4+6)}$;
- ...
- $N_q(k^m) \approx \hbar^m \cdot k^{-(2+4+\ldots+2m)}$.

Here,

$$2 + 4 + \ldots + 2m = 2 \cdot (1 + 2 + \ldots + m) = 2 \cdot \frac{m \cdot (m+1)}{2} \approx m^2,$$

so $N_q(k^m) \approx \hbar^m \cdot k^{-m^2}$.

To get the expression for $N_q(\varepsilon)$, we need to find the value m for which $k^m = \varepsilon$. This value is $m \sim \ln(\varepsilon)$. Substituting this value into the above formula, we conclude that

$$N_q(\varepsilon) \sim \exp(\alpha \cdot \ln^2(\varepsilon))$$

for some real value α.

How does this formula translate into a speedup? As we have mentioned, in modern physics, the speeds of all the processes are limited by the speed of light. Thus, the time of performing a single computational operation on a processing unit of linear size ε is $\geq \varepsilon/c$. So, to make computers faster, we need to make computational units smaller. The linear size ε of the computational unit can thus serve as the description of a technological level.

To find out how much speedup we can achieve by using quantum space-time, for the same technological level ε, we need to compare:

- parallel computations in non-quantum space-time, and
- parallel computations in quantum space-time.

In non-quantum space-time, we have units of volume ε^3, so within the region of volume V_0 we can fit

$$N_n(\varepsilon) \sim \frac{V_0}{\varepsilon^3} \sim \varepsilon^{-3}$$

such units.

In quantum space-time, the number $N_q(\varepsilon)$ of computational units is equal to $N_q(\varepsilon) \sim \exp(\alpha \cdot \ln^2(\varepsilon))$. In order to describe N_q in terms of N_n, we need to express ε in terms of N_n and then substitute the resulting expression into the formula for N_q. From the formula for N_n, we conclude that $\varepsilon \sim N_n^{-1/3}$. Therefore, $\ln(\varepsilon) \sim \ln(N_n)$. Substituting these expressions for ε and $\ln(\varepsilon)$ into the formula for N_q, we conclude that

$$N_q \sim \exp(\beta \cdot \ln^2(N_n)) = N_n^{\beta \cdot \ln(N_n)}.$$

This expression grows faster than any polynomial. Hence, in quantum space-time, parallelization can potentially leads to super-polynomial speedup of computations.

Comments.

- This result was first announced in [9].
- From the practical viewpoint, parallelization it is not sufficient to guarantee the computational speedup: in order to gain speed by parallelization, one must have a strategy how a particular input is distributed among multiple processors. Our suggestion is to follow a strategy outlined – for a similar situation – in [4].

Acknowledgments

This work was supported in part by two grants from the US National Science Foundation (NSF):

- Cyber-ShARE Center of Excellence (HRD-0734825),
- Computing Alliance of Hispanic-Serving Institutions CAHSI (CNS-0540592),

and by Grant 1 T36 GM078000-01 from the US National Institutes of Health (NIH).

The authors are thankful to all the participants of the 7th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences (Las Cruces, New Mexico, April 3, 2010) for valuable discussions.

References

[1] R. Feynman, R. Leighton, and M. Sands, *The Feynman Lectures on Physics*, Addison Wesley, Boston, Massachusetts, 2005.

[2] A. M. Finkelstein and V. Kreinovich, "The singularities in quantum cosmology", *Astrophysics and Space Science*, 1987, Vol. 137, No. 1, pp. 73–76.

[3] V. Kreinovich and A. M. Finkelstein, "Towards applying computational complexity to foundations of physics", *Notes of Mathematical Seminars of St. Petersburg Department of Steklov Institute of Mathematics*, 2004, Vol. 316, pp. 63–110; reprinted in *Journal of Mathematical Sciences*, 2006, Vol. 134, No. 5, pp. 2358–2382.

[4] V. Kreinovich and M. Margenstern, "In some curved spaces, one can solve NP-hard problems in polynomial time", *Notes of Mathematical Seminars of St. Petersburg Department of Steklov Institute of Mathematics*, 2008, Vol. 358, pp. 224–250; reprinted in *Journal of Mathematical Sciences*, 2009, Vol. 158, No. 5, pp. 727–740.

[5] C. W. Misner, K. S. Thorne, and J. A. Wheeler, *Gravitation*, Freeman, San Francisco, 1973.

[6] 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.

[7] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, Massachusetts, 2000.

[8] C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, San Diego, California, 1994.

[9] M. Zakharevich and V. Kreinovich, "Computation in quantum space-time can lead to a super-polynomial speedup", in: *Abstracts of the 7th Joint UTEP/NMSU Workshop on Mathematics, Computer Science, and Computational Sciences*, Las Cruces, New Mexico, April 3, 2010.