The Complexity of Iterated Reversible Computation

David Eppstein
University of California, Irvine

15th Latin American Theoretical Informatics Symposium
(LATIN 2022)
Guanajuato, Mexico, November 2022

Introduction: Jeweled reflection

Interior reflections in jewels

“Brilliant cut” diamonds are designed so that most light entering the table (top flat surface) reflects many times before exiting the table

[Fatimazt 2018]

Q [Stefan Langerman]: How can we simulate this on a computer?

Obvious method: Trace reflected light rays until they exit

But there may be a very large number of reflections

Maybe there is some less-obvious speedup?

How light reflects in a jewel

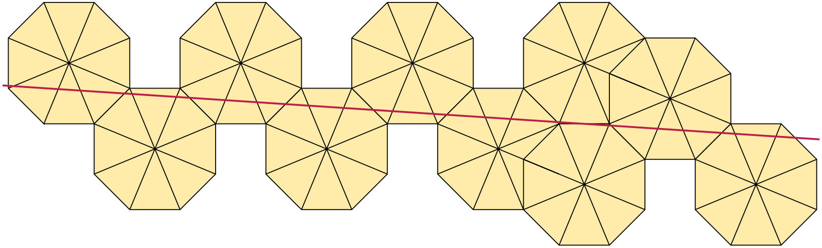
When a light ray hits a face of the diamond:

- It enters the interior, refracted so that its angle to the face is steep, greater than the critical angle θ
- As long as it hits other faces at shallow angles $< \theta$, it reflects from them at the same angle ("total internal reflection")
- It will exit the jewel through the first face that it hits at another steep angle $> \theta$

A toy version of the problem (still unsolved)

Instead of convex polyhedron, suppose jewel is isosceles triangle

Equivalent to unfold the reflected ray into a straight line
through reflected copies of the triangle



Behavior depends on its angles and on critical angle

Can we find exit point, number of reflections, or exit angle faster
than performing each reflection step-by-step?

Abstraction: reversibility

The reflection problem is *reversible*: If we know the position and direction of the light ray at any point, we can trace its reflections backwards step-by-step in the same way to find its entrance point.

Each reflection step preserves all of the information from the input

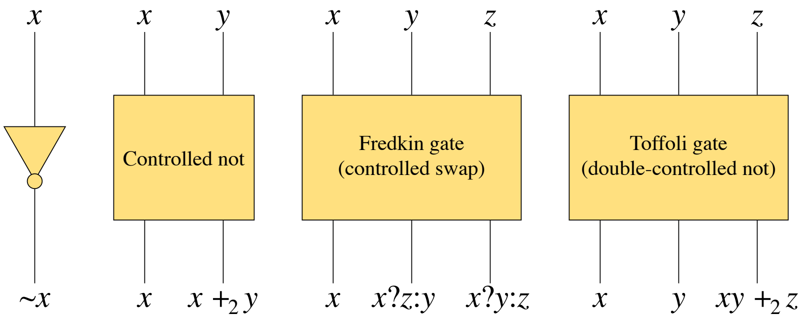
Individual reflections are easy-to-compute; it's only their composition that makes things difficult

Can computing compositions of many easy reversible steps be hard, or are such problems always easy?

Reversible computation

Reversible logic

Boolean circuit made of reversible gates

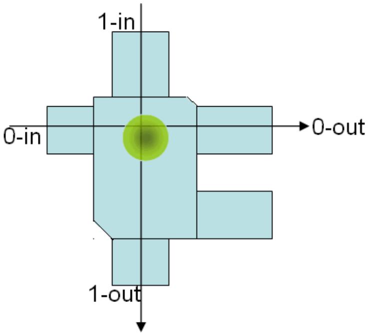


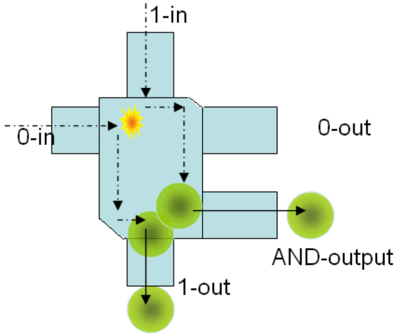
Each gate has same number of inputs and outputs
and maps each input to a unique output

(No fan-outs: that would be a gate with more outputs than inputs)

Billiard-ball computing

Can simulate reversible gates by axis-parallel balls and fixed reflectors with elastic collisions



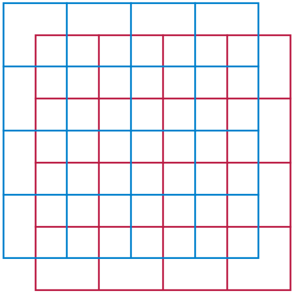


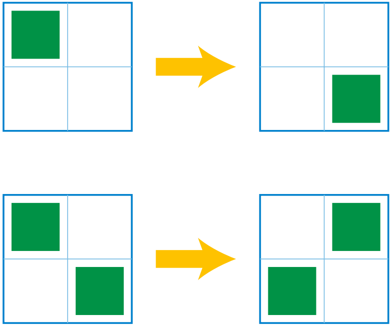
[Newcomp 2007]

Reversible cellular automata

Grid of cells with finite states, local rule for synchronous update

Block reversible CA: in alternating steps, partition cells into 2×2 blocks and apply the same bijection to the states in each block





Example: Diagonally moving particle and collision in simulation of billiard-ball computer (other types of 2×2 block do not change)

Why?

These models seem difficult to build as physical computers, and difficult to program once built, so why should we care?

Original motivation: can compute with arbitrarily low energy cost

In contrast, there is a physical lower bound on the energy of a single step of conventional logic, the Landauer limit $k_{\mathrm{B}} T \ln 2$

[Landauer 1961; Bennett 1982; Vitányi 2005]

Bigger current motivation: Quantum devices must be reversible

Deterministic reversible logic helps understand quantum logic

[Peres 1985; 't Hooft 1988; Deutsch 1989]

Computational complexity

We need some padding

Even very simple reversible functions cannot be calculated by reversible circuits

Example: map n-bit binary number x to $-x$

Swaps $2^n - 2$ pairs of inputs

Leaves two inputs (zero and all-ones) unchanged

This is an odd permutation of the 2^n input values

Circuits without n-to-n gates can only compute even permutations

With padding, everything can be computed

Expand your favorite algorithm into a conventional logic circuit

Replace each gate by one with additional zeros (padding) as input, and additional outputs that copy the real inputs (“garbage”, unused by the rest of the computation)

Copy the result then reverse the circuit to zero out the garbage

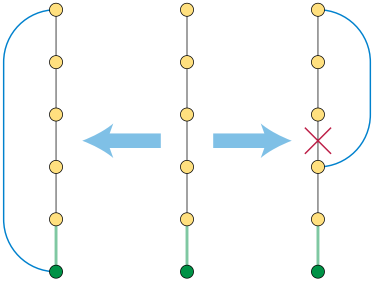
$\Rightarrow$ polynomial time = reversible polynomial time

But what about space complexity?

Thomason's lollipop algorithm

In a 3-regular graph with a Hamiltonian path starting from a fixed edge:

- Extend the path by one more edge ⇒ cycle or "lollipop"
- Remove an edge from lollipop to get another path



Connects paths and cycles into state space ⇒ follow sequence of states from any Hamiltonian cycle to find another [Thomason 1978]

Hardness of following sequences of states

Abstraction of the lollipop algorithm:

- We have a subroutine for finding neighbors in an unknown graph of degree ≤ 2, with an exponential number of vertices
- We are given a vertex of degree one, which must belong to a path in the graph
- Goal: find the other endpoint of the path

Known: Complete for $\mathsf{FP}^{\mathsf{PSPACE}}$ (functional analogue of PSPACE)

[Bennett 1984; Papadimitriou 1994; Lange et al. 2000]

From state sequences to functional iteration

Choose large N (larger than number of states)

Replace state with triple (state,direction,counter)

Bijection on triples:

- If counter is nonzero, increment it mod N
- Otherwise, move state one step in given direction
- If you reach a degree-one state, reverse direction and increment counter

Walks back and forth along paths of states, delay N at each end

For any state on a path, $f^{(N)}$(state) is endpoint in given direction

Hardness of iterated bijections

Because hard path problems can be turned into iterated bijection, the following problems are all complete for $\mathsf{FP}^{\mathsf{PSPACE}}$:

- Computing $f^{(N)}(x)$, for a polynomial-time computable bijection (described by an algorithm and a unary time bound)
- Computing $f^{(N)}(x)$, for a function described by the behavior of a reversible logic circuit
- Computing the state of a billiard-ball computer after N steps
- Computing the state of the BBM reversible cellular automaton with periodic boundary conditions after N steps

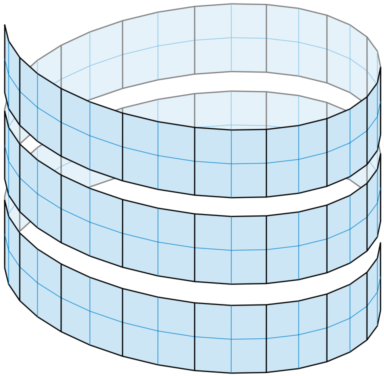
The logic circuit hardness result requires some padding tricks to keep the garbage bits from piling up; see [Jacopini et al. 1990]

Hardness of 1d reversible cellular automata

Also hard: N steps of 1d reversible CA with periodic boundary

Known universal 1d RCA use unbounded space [Morita 2007]

Instead, simulate 2d BBM with helical boundary, by interspersing BBM steps with steps that slide data around helix, synchronized by firing-squad RCA [Imai and Morita 1996]



Evades proof that RCA simulation cannot lower dimension [Hertling 1998] by taking non-constant time to simulate each step of 2d CA

Dynamical systems

Dynamical systems

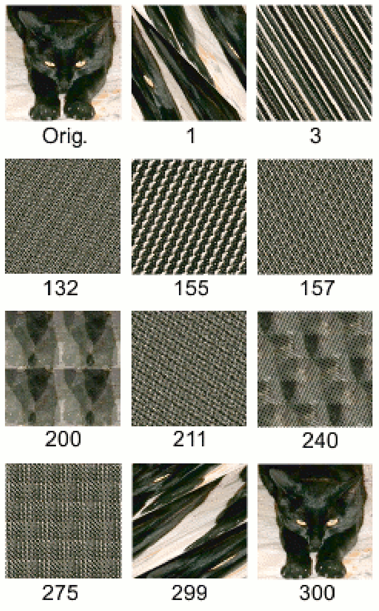
Mathematical study of iterated bijections

Even very simple bijections (e.g. piecewise linear) can have complex behavior when iterated

Example: Arnold's cat map

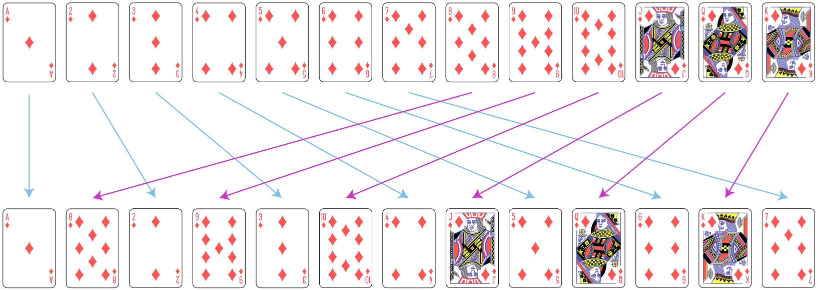
$(x, y) \mapsto (2x + y, x + y) \bmod 1$

[Dyson and Falk 1992]



[Rocchini 2006]

Shuffling cards



[Fomin 2017]

Perfect riffle shuffles (“Faro shuffles”) are piecewise linear

On a deck of 2^k cards, they map position i to $\mathrm{rot}(i)$, where rot is one-step bitwise left-rotation of k-bit binary numbers

Iterated piecewise linear transforms are hard

Proof idea: Use card tricks

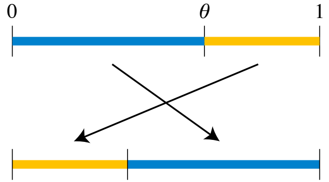
- Shuffle $\Rightarrow$ rotate binary numbers
- Cut deck into $O(1)$ packs and reorder them $\Rightarrow$ any reversible operation on the $O(1)$ high-order bits
- Sequences of shuffles and cuts $\Rightarrow$ any reversible circuit
- Pipeline sequence into single hard-to-iterate one-dimensional piecewise linear transformation

[Andersen 2015]

Easier to iterate: interval exchange

Partition an interval into subintervals and permute them

$\Rightarrow$ piecewise-translational bijection



We are interested in intervals of *integers*
and integer-preserving transformations of them:

"Integer interval exchange transformation"

Description size = # subintervals $\times$ $\log_2$(interval length)

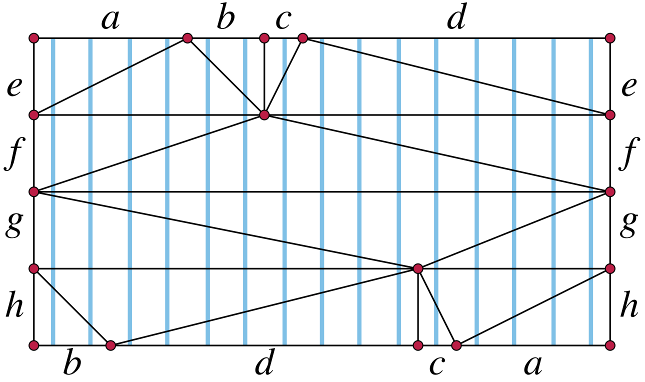
From interval exchange to topology

Triangulated rectangle + glued boundary + vertical "normal curves"

Top: intervals of exchange transformation

Bottom: their permutation

Middle: single edge of triangulation

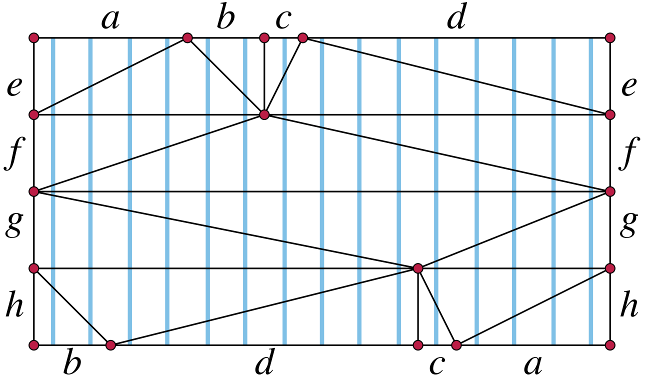


$\tau^{(N)}(x)$: start at position x on middle edge, then travel vertically for height $\times$ $2N$ steps along normal curve

Interval exchange transformation algorithm

To compute $\tau^{(N)}(x)$ for interval exchange transformation τ:

Transform transformation into normal curves of triangulated surface



Compute what happens if you travel k steps along a normal curve using algorithms from [Erickson and Nayyeri 2013]

Translate everything back into terms of original problem

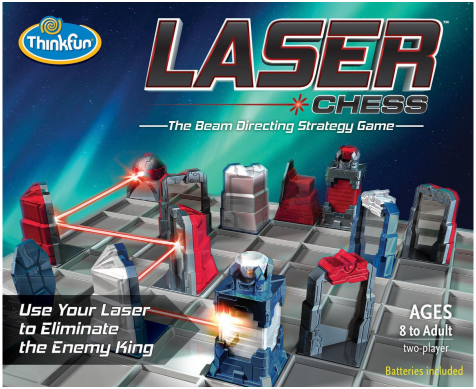
Reflection revisited

The general reflection problem

[Morin 2019]

To render a scene involving repeated mirror reflections, do we have to follow light rays one reflection at a time, or can we use shortcuts?

A drastic (over)simplification


ThinkFun
LASER CHESS
The Beam Directing Strategy Game
Use Your Laser to Eliminate the Enemy King
AGES 8 to Adult
two-player
Batteries included

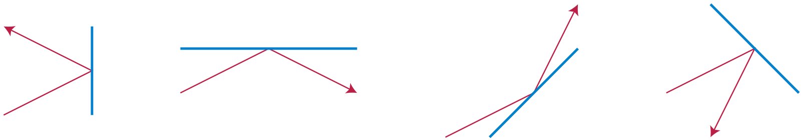
2D environment of axis-parallel and slope-± 1 mirrors and obstacles

All endpoints have integer coordinates

Viewpoint has integer coordinates and rational slope

Why these restrictions?

Reflected slopes are limited to a small set!

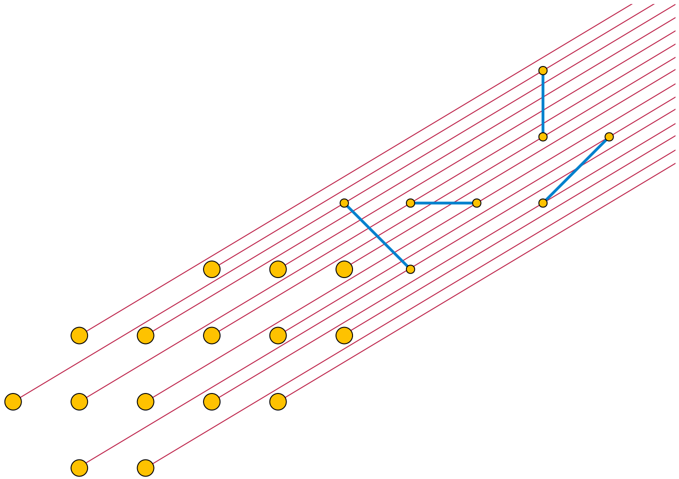


$$s \to \{\pm s, \pm \frac{1}{s}\}$$

... and the reflected ray stays on a line through integer points

How fixed-slope integer-origin rays hit a mirror

Points of intersection are non-integer but evenly spaced

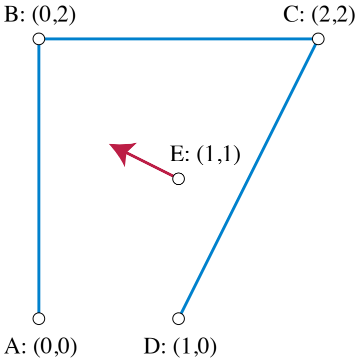


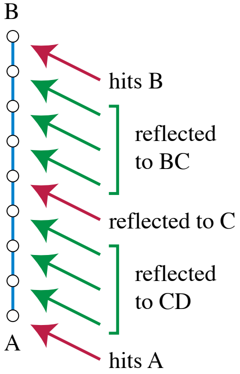
We can compute how many from the slope of the ray and mirror

From mirrors to interval exchange

For each mirror and each of $O(1)$ possible incoming slopes:

- Mirror + slope has discrete set of reflection points
- Concatenate these sets of reflection points for all mirrors and slopes into a single big integer interval
- Form subintervals for where a reflected ray will go next

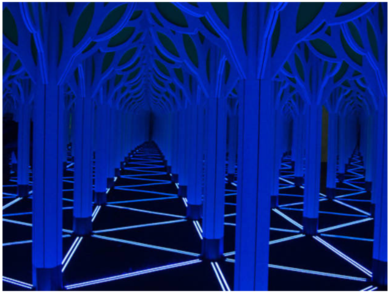




But where do the blocked/escaping rays go?

An interval exchange transformation must be a bijection

Make a "trap": part of the transformation that maps long subinterval to itself + small shift (like two parallel mirrors) so it takes huge number of steps to escape



[Kevdog686 2019]

Send rays that have no next reflection into start of trap

Give preimages from end of trap to rays that need one

Transformed problem, so far

Turn input scene and line of sight into an
equivalent integer interval exchange transformation τ

- # intervals = linear in # scene features
- length of interval = polynomial in scene coordinates
- initial ray = initial value x for transformation
- every ray is trapped in $\leq N$ steps, for polynomial N
- trap takes more than N steps to escape

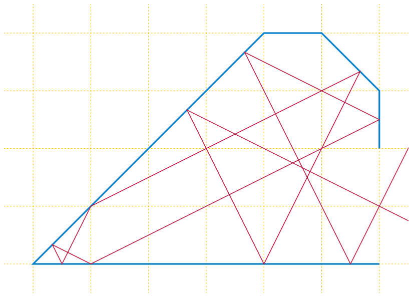
$\Rightarrow$ We just need to compute $\tau^{(N)}(x)$ and decode it!

. . . we already know how to do this using computational topology

Octagonal reflection, solved

Given a 2D octagonal scene and a rational ray

We can find the eventual fate of the ray (the obstacle and angle that it hits, or a ray it escapes along) in polynomial time



Weakly polynomial: Depends on # bits of input coordinates, not just on # mirrors and obstacles in input scene

But number of reflections can be exponential in the same quantity, so this is much better than step-by-step simulation

Conclusions and references

Conclusions

Studied a general class of problems involving N-fold iteration of polynomial-time bijections

Includes problems in graph algorithms, circuit complexity, cellular automata, dynamical systems, computational topology, and computer graphics

Many are hard

Integer exchange and octagonal reflection have polynomial time algorithms

Sources

D. Eppstein
The Complexity of Iterated Reversible Computation
Preprint, 2022

D. Eppstein
Reflections in an Octagonal Mirror Maze
Canadian Conference on Computational Geometry, 2022

References and image credits, I

David Mejlhede Andersen. David Mejlhede Performing the Sybil Cut using Madison Hustlers from Daniel Madison and Ellusionist. CC-BY-SA image, December 9 2015. URL `https://commons.wikimedia.org/wiki/File:David_Mejlhede_performing_the_Sybil_Cut.jpeg`.

Charles H. Bennett. The thermodynamics of computation – a review. *International J. Theoretical Physics*, 21(12):905–940, 1982. doi: 10.1007/bf02084158.

Charles H. Bennett. [TIME, SPACE] $(T, S) \subseteq$ REVERSIBLE [TIME, SPACE] $(T^{1.585}, S \log T)$. Technical report, IBM Yorktown Heights, 1984. As cited by [Papadimitriou 1994].

David Elieser Deutsch. Quantum computational networks. *Proceedings of the Royal Society A*, 425(1868):73–90, 1989. doi: 10.1098/rspa.1989.0099.

References and image credits, II

Freeman J. Dyson and Harold Falk. Period of a discrete cat mapping. *American Mathematical Monthly*, 99(7):603–614, 1992. doi: 10.2307/2324989.

Jeff Erickson and Amir Nayyeri. Tracing compressed curves in triangulated surfaces. *Discrete & Computational Geometry*, 49 (4):823–863, 2013. doi: 10.1007/s00454-013-9515-z.

Fatimazt. Daimond de mer. CC-BY-SA image, August 3 2018. URL `https://commons.wikimedia.org/wiki/File:Daimond.jpg`.

Dmitry Fomin. English pattern playing cards deck. Public-domain image, 2017. URL `https://commons.wikimedia.org/wiki/File:English_pattern_playing_cards_deck.svg`.

Peter Hertling. Embedding cellular automata into reversible ones. In *Unconventional models of computation (Auckland, 1998)*, Springer Series in Discrete Mathematics and Theoretical Computer Science, pages 243–256. Springer-Verlag, 1998.

References and image credits, III

Katsunobu Imai and Kenichi Morita. Firing squad synchronization problem in reversible cellular automata. *Theoretical Computer Science*, 165(2):475–482, 1996. doi: 10.1016/0304-3975(96)00016-3.

G. Jacopini, P. Mentrasti, and G. Sontacchi. Reversible Turing machines and polynomial time reversibly computable functions. *SIAM Journal on Discrete Mathematics*, 3(2):241–254, 1990. doi: 10.1137/0403020.

Kevdog686. A picture of the inside of the mirror maze showcasing its infinite looking repetition. CC-BY-SA image, May 27 2019. URL `https://commons.wikimedia.org/wiki/File:Mirror_Maze_in_the_Museum_of_Science%2BIndustry_of_Chicago.jpg`.

R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191, 1961. doi: 10.1147/rd.53.0183.

References and image credits, IV

Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible space equals deterministic space. *J. Computer and System Sciences*, 60(2, part 2):354–367, 2000. doi: 10.1006/jcss.1999.1672.

Basile Morin. Street crowd reflecting in the polyhedral mirrors of the station Tokyu Plaza Omotesando, Harajuku, Tokyo, Japan. CC-BY-SA image, June 16 2019. URL `https://commons.wikimedia.org/wiki/File:Street_crowd_reflecting_in_the_polyhedral_mirrors_of_the_station_Tokyu_Plaza_Omotesando,_Harajuku,_Tokyo,_Japan.jpg`.

Kenichi Morita. Simple universal one-dimensional reversible cellular automata. *Journal of Cellular Automata*, 2(2):159–165, 2007.

Newcomp. Toffoli BilliardBall. CC-BY-SA image, February 4 2007. URL `https://commons.wikimedia.org/wiki/File:Toffoli_BilliardBall.gif`.

References and image credits, V

Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Computer and System Sciences*, 48(3):498–532, 1994. doi: 10.1016/S0022-0000(05)80063-7.

Asher Peres. Reversible logic and quantum computers. *Physical Review A*, 32(6):3266–3276, 1985. doi: 10.1103/PhysRevA.32.3266.

Claudio Rocchini. Arnold's cat map sample (Chaos theory). CC-BY-SA image, November 8 2006. URL https://commons.wikimedia.org/wiki/File:Arnold_cat.png.

Gerard 't Hooft. Equivalence relations between deterministic and quantum mechanical systems. *J. Statistical Physics*, 53(1-2): 323–344, 1988. doi: 10.1007/BF01011560.

A. G. Thomason. Hamiltonian cycles and uniquely edge colourable graphs. *Annals of Discrete Mathematics*, 3:259–268, 1978. doi: 10.1016/S0167-5060(08)70511-9.

References and image credits, VI

Paul M. B. Vitányi. Time, space, and energy in reversible computing. In Nader Bagherzadeh, Mateo Valero, and Alex Ramírez, editors, *Proc. 2nd Conf. on Computing Frontiers (CF 2005)*, pages 435–444. ACM, 2005. doi: 10.1145/1062261.1062335.