

Universality and Cellular Automata

K. Sutner

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
sutner@cs.cmu.edu

Abstract. The classification of discrete dynamical systems that are computationally complete has recently drawn attention in light of Wolfram’s “Principle of Computational Equivalence”. We discuss a classification for cellular automata that is based on computably enumerable degrees. In this setting the full structure of the semilattice of the c.e. degrees is inherited by the cellular automata.

1 Intermediate Degrees and Computational Equivalence

One of the celebrated results of recursion theory in the 20th century is the positive solution to Post’s problem: there are computably enumerable sets whose Turing degree lies strictly between \emptyset , the degree of any recursive set, and \emptyset' , the degree of the Halting set or any other complete computably enumerable set. The result was obtained independently and almost simultaneously by R. M. Friedberg and A. A. Muchnik, see [8, 14]. The method used in their construction of an intermediate degree is remarkable since it departs significantly from earlier attempts by Post and others to obtain such degrees by imposing structural conditions such as simplicity, hyper-simplicity or hyper-hyper-simplicity on the corresponding c.e. sets, see [17, 11] for background information. The conditions are chosen so as to clearly rule out decidability and the hope was they also might enforce incompleteness. Of course, a non-trivial existence proof is required for this approach to succeed. Unfortunately, these attempts failed quite dramatically. For example, it was shown by Dekker that there is a hyper-simple set in every non-recursive computably enumerable degree and hence in particular in the complete degree.

By contrast, the so-called priority method used in the Friedberg-Muchnik construction builds two c.e. sets A and B whose only tangible property is that they are incomparable with respect to Turing reductions. The method generalizes easily to a construction of infinitely many incomparable sets. Alas, the sets constructed via priority arguments appear somewhat ad hoc and artificial. It is therefore tempting to search for “natural” examples of intermediate degrees,

examples that would presumably arise as a side-effect of a less complicated construction. By natural we here mean that the generating device should admit a very simple description as opposed to, say, invariance under automorphisms of the semilattice of c.e. degrees. Of course, there are well-known results to the effect that every c.e. degree appears in a certain mathematical context. For example, all c.e. sets are Diophantine and can thus be defined by an integer polynomial. Similarly, every c.e. set is Turing equivalent to a finitely axiomatizable theory and word problems in finitely presented groups may have arbitrary c.e. degree. But the point here is to obtain a specific example of an intermediate degree using a reasonably simple mechanism to do so. For example, an elementary cellular automaton would certainly provide an indisputably natural example for an intermediate degree. A candidate for such an elementary cellular automaton might be rule 30, see the comments in section 5.

However, it is not clear at present whether there is much hope to find such examples. Indeed, in [28] Wolfram introduces his “Principle of Computational Equivalence” (PCE) which suggests, among other things, that the search is futile: Wolfram asserts that a certain class of computational processes obeys a 0/1 law: these processes are either decidable or already computationally universal. The evidence that Wolfram adduces for this principle is directly relevant to the search of natural examples: a large collection of simulations on various discrete dynamical systems such as Turing machine, register machines, tag systems, rewrite systems, combinators, and cellular automata. It is pointed out in chapter 3 of [28], that in all these classes of systems there are examples that exhibit exceedingly complicated behavior (and presumably even computational universality) even when the system in question is surprisingly small and has a very short description.

The reference contains a particularly striking example for a universal system that nonetheless has a very simple description: elementary cellular automaton rule number 110. The local map of this automaton is given by the ternary boolean function $\rho(x, y, z) = (\bar{x} \wedge y \wedge z) \oplus y \oplus z$ where \oplus denotes exclusive or. It requires significant effort to show that, using fairly complicated and large segments of bits as the basic building blocks, it is possible to simulate cyclic tag systems on this cellular automaton, thus proving universality. There is a noteworthy difference between this and earlier examples of computationally universal cellular automata: the required configurations for this argument do not have finite support but are ultimately periodic in both spatial directions. In fact, it is not hard to see that rule 110 produces no interesting orbits on configurations with finite support.

While we are mostly interested in cellular automata as possible sources of natural intermediate degrees, we will first describe the problem in the slightly more general setting of an effective dynamical system. As a general purpose tool to measure the complexity of such a system we adapt M. Davis’s notion of universality of Turing machines, see [5, 6], to avoid coding conventions. We then

discuss how systems of intermediate degree of complexity appear in various contexts. Needless to say, all of these results are universal rather than specific: they are similar to Matiyasevic’s characterization of the Diophantine equations mentioned above in that they show that all c.e. degrees appear in some context but do not produce concrete or natural examples. Proofs for the results quoted in this paper as well as technical details can be found in the references.

2 Effective Dynamical Systems, Universality and Classification

We consider effective dynamical systems of the form $\mathcal{C} = \langle \mathcal{C}, \rightarrow \rangle$. Here \mathcal{C} is the space of *configurations*, and \rightarrow is the “next configuration” relation. We write $\overset{*}{\rightarrow}$ for the transitive reflexive closure of \rightarrow . The configurations are required to be finitary objects such as words and admit natural codings as integers. The relation \rightarrow has to be primitive recursive on the (codes of the) configurations. In fact, in all relevant examples \rightarrow is primitive recursive uniformly in a parameter that describes the particular instance of the system. E.g., for cellular automata the next configuration relation is primitive recursive uniformly in the local map of the automaton. Elementary cellular automata in particular can be parameterized by an 8-bit rule number. It is clear that the systems discussed in Wolfram’s book all fit this general pattern.

Computational universality is traditionally defined in terms of simulations. First, one fixes a coding and decoding function $\alpha : \mathbb{N} \rightarrow \mathcal{C}$ and $\beta : \mathcal{C} \rightarrow \mathbb{N}$. Second, one adopts a notion of termination for the effective dynamical system so that an orbit may or may not lead to a “halting” configuration. It has to be easily decidable, say, primitive recursive, whether a configuration is halting. If a halting configuration appears in the orbit of $\alpha(n)$ let Y be the first such configuration and interpret $\beta(Y)$ as the output of the computation by \mathcal{C} on input n . It is clear that \mathcal{C} computes only partial recursive functions and it is natural to consider the system complete if it can compute all such functions.

While this approach is perfectly adequate for Turing machines or register machines it becomes a bit more problematic in the realm of cellular automata. There is no clear natural notion of termination here and even the coding functions are not so obvious. We therefore sidestep the issue of simulations entirely and instead adapt Davis’s definition for universality of a Turing machine. In [6] Davis suggests that a more robust measure for the complexity of a Turing machine is the Turing degree of the collection of its instantaneous descriptions that lead to a halting configuration. For a Turing machine M let $ID_M = \{ I \mid I \overset{*}{\rightarrow} J, J \text{ halting} \}$. Then the machine is *Davis-universal* if ID_M is complete c.e.. Thus, there are no coding functions that might contribute to the apparent complexity of the machine. It is easy to see that any classically universal Turing machine is also Davis-universal; however, the opposite is false:

a Turing machine that erases its tape before halting has trivial input/output behavior but may still be Davis-universal. Surprisingly, it was shown by Davis that any total recursive function can be computed by a *stable* Turing machine: all its instantaneous descriptions lead to a halting configuration. Thus, a stable Turing machine is trivial in the sense of Davis-universality: ID_M comprises all instantaneous descriptions and is trivially decidable.

In the context of an effective dynamical system \mathcal{C} it is thus natural to consider the *complete orbit*

$$\text{Orb}_{\mathcal{C}} = \{ (X, Y) \mid X \xrightarrow{*} Y \}$$

which is c.e. given our constraints on the next configuration relation. We can then use the Turing degree of $\text{Orb}_{\mathcal{C}}$ as a measure of the complexity of \mathcal{C} . Alternatively, we can interpret the degree measure as a decision problem, the *Reachability Problem* for \mathcal{C} : given two configurations X and Y we wish to determine whether Y lies in the orbit of X : is $X \xrightarrow{*} Y$?

Let us now focus on cellular automata. Since we insist on configurations being finitary we need to constrain the full Cantor space Σ^∞ of all biinfinite words over alphabet Σ . To obtain a reasonable class of configurations $\mathcal{C} \subseteq \Sigma^\infty$ one should insist that \mathcal{C} is shift-invariant and dense. Furthermore, \mathcal{C} must be closed under continuous shift-invariant maps (i.e. the global maps of cellular automata). Lastly, in order to obtain a reasonable image of the dynamics of the map on the whole space we insist on *reflection*: whenever a configuration $X \in \mathcal{C}$ has a predecessor under \rightarrow in Σ^∞ then it also has a predecessor in \mathcal{C} . The classical choice for such a space of configurations is \mathcal{C}_f , the collection of all configurations with finite support (to obtain reflection one has to either insist on quiescence of the local map or interpret the notion of finite support appropriately). Another possibility is to consider spatially periodic configurations of the form ${}^\omega u^\omega$. These configurations are somewhat special in that they correspond to finite cellular automata with periodic boundary conditions and all orbits here are necessarily finite. On the other hand, the largest such configuration space is the collection of all recursive configurations, see [20] for a proof that reflection holds in this case.

However, the Cook-Wolfram proof of the universality of elementary cellular automaton rule 110 suggests to consider a much more narrow class of configurations. To this end, define a configuration to be *almost periodic* if it has the form $X = {}^\omega u v v^\omega$ where u , v and w are all finite words. Then the class \mathcal{C}_{ap} of all almost periodic configurations has all the properties from above. Note that reflection does not hold for configurations of the more restricted form ${}^\omega u v u^\omega$. Furthermore, it is not clear how to transfer the universality argument for rule 110 into this setting. More precisely, the Cook-Wolfram argument uses the infinitely many copies of u in ${}^\omega u v v^\omega$ to produce timing signals whereas the copies of v encode the cyclic tag system. Both together operate on the center part w where the actual simulation of the tag system takes place.

Given a space \mathcal{C} of suitable configurations we can define the *degree classification* for cellular automata as follows. For any c.e. degree \mathbf{d} let

$$\mathbb{C}_{\mathbf{d}} = \{ \rho \mid \deg(\text{Orb}_{\rho}) = \mathbf{d} \}.$$

where ρ denotes the local map of the cellular automaton. We note that this classification does not distinguish between the first three levels of the Wolfram classification in its formalization by Culik and Yu: these three levels are subsumed by \mathbb{C}_{\emptyset} . On the other hand, class \mathbb{C}_{\emptyset} comprises all computationally universal cellular automata. The following hierarchy theorem is established in [21, 23] over \mathcal{C}_f .

Theorem 1. *The degree classes $\mathbb{C}_{\mathbf{d}}$ are non-empty for every c.e. degree \mathbf{d} .*

The construction uses a simulation of a Turing machine that recognizes an c.e. set W of degree \mathbf{d} . As in the case of Davis-universality we have to contend with unintended instantaneous descriptions, i.e., instantaneous descriptions that do not occur in any actual computation of the Turing machine. Since it is undecidable whether, say, a state of the Turing machine appears in a computation this requires a somewhat more careful construction than usual. The notion of stability can be relaxed in this context to mean that unintended configurations only produce decidable orbits and thus do not alter the degree of Orb_{ρ} . Incidentally, in his original paper Davis uses a syntactic normal form for computable functions rather than a direct modification of Turing machines, see also [15] and [1] for similar arguments. A suitably modified Turing machine can then be simulated by a one-dimensional cellular automaton to establish the theorem, see [23] and [18] for details. The latter reference in particular contains a detailed discussion of the coding issues.

As a consequence of this result there is little hope to obtain a taxonomy of cellular automata based on a few simple classes. For example, it follows from Sack's density theorem that for any two cellular automaton ρ_1 and ρ_2 such that $\text{Orb}_{\rho_1} <_T \text{Orb}_{\rho_2}$ there exists a third cellular automaton σ of strictly intermediate complexity: $\text{Orb}_{\rho_1} <_T \text{Orb}_{\sigma} <_T \text{Orb}_{\rho_2}$. It is well-known that the Σ_1 -theory of the semilattice of c.e. degrees is decidable. However, the reason for this decidability result lies in the fact that any countable partial order can be embedded into the semilattice so that the relative computational strength of cellular automata is indeed arbitrarily complicated. The full theory of the semilattice of c.e. degrees is known to be undecidable, see [9]; in fact it is extraordinarily complicated: its degree is $\emptyset^{(\omega)}$.

While Reachability deals with forward orbits, another classical decision problem for cellular automata is the existence of predecessors: given Y , is there a configuration X such that $X \rightarrow Y$? Configurations that do not admit a predecessor are often referred to as a Garden-of-Eden. It is easy to see that for finite or almost periodic configurations the existence of a predecessor is easily solvable in polynomial time. However, in the two-dimensional case the existence of an arbitrary

predecessor configuration, given a finite target configuration, is co-c.e.-complete. The same problem is c.e.-complete for finite configurations and one can show that a suitable choice of cellular automaton will produce a predecessor problem of arbitrary c.e. degree, see [21].

More complicated predecessor problems for dimension one appear when we enlarge the class of admissible configurations to all recursive configurations. In this case it is co-c.e.-complete to determine whether a given configuration has a predecessor for any non-surjective cellular automaton, see [23]. Of course, surjectivity itself is easily testable in polynomial time for one-dimensional cellular automata.

3 Testing Complexity

The heuristic classification of one-dimensional cellular automata due to Wolfram [26] is based on the visual inspection of a segment of the orbits of the automaton. For sufficiently simple cellular automata the classification is quite compelling, see the many examples in Wolfram's book. However, any attempt to formalize this process in general seems to lead to strong undecidability. For example, Culik and Yu translated the Wolfram classes into the following categories: all configurations evolve to a fixed point, all configurations evolve to a periodic configuration and all configurations have decidable orbits, plus the class of all remaining cellular automata. These classes are shown to be undecidable in [4] where the underlying space of configurations here is \mathcal{C}_f . We mention reference [1] in passing for another objection to the Wolfram classification. Closer inspection of the low classes shows the following.

Theorem 2. *It is Π_2 -complete to check whether a finite configuration evolves to a fixed point. The same holds true for evolution to a periodic configuration.*

Spatially periodic configurations of course always evolve to periodic configurations but even in this case it is co-c.e.-complete to test whether a fixed point is ultimately reached. Likewise it is Σ_2 -complete to test whether the inevitable limit cycle has length $O(n^k)$ for some fixed k . Here n denotes the length of the periodic configuration, see [19].

Unsurprisingly, it is even more difficult to determine the type of a cellular automaton in the degree classification.

Theorem 3. *For any c.e. degree \mathbf{d} it is $\Sigma_3^{\mathbf{d}}$ -complete to determine whether a cellular automaton belongs to class $\mathbb{C}_{\mathbf{d}}$.*

Similar results hold, *mutatis mutandis*, for the analogous cumulative hierarchies $\mathbb{C}_{\leq \mathbf{d}}$ and $\mathbb{C}_{\geq \mathbf{d}}$, see [23]. For example, $\mathbb{C}_{\leq \mathbf{d}}$ is $\Sigma_3^{\mathbf{d}}$ -complete for all $\mathbf{d} < \emptyset'$, but

$\mathbb{C}_{\leq \emptyset'}$ comprises all cellular automata and is thus trivial. It follows that it is Σ_3 -complete to determine whether all orbits of a cellular automaton are decidable. However, testing whether the orbits are c.e.-complete is a Σ_4 -complete task. In light of these undecidability results it would be desirable to develop a collection of sufficient conditions for universality. For example, for certain variants of the Game-of-Life there appears to be hope to identify the key mechanisms that make some of these automata universal, see [7]. Needless to say, it will be much harder to find sufficient criteria for properties that prevent universality without trivializing the orbits, but see the comments in section 5.

Another source of undecidability is the choice of appropriate backgrounds ${}^\omega uv^\omega$ in an almost periodic configuration ${}^\omega uuv^\omega$. As an example, consider again the universal elementary cellular automaton rule 110. If both u and v have length 1 then the orbit of any almost periodic configuration ${}^\omega uuv^\omega$ is trivially decidable. It is not hard to check that the same is true for slightly larger background patterns u and v . On the other hand, for sufficiently long background patterns the orbits become undecidable. There is no algorithmic way to determine the threshold between the two types of behavior.

Theorem 4. *Given a cellular automaton it is undecidable whether orbits on almost periodic configurations are undecidable for sufficiently long background patterns.*

For rule 110 it is the case that c.e.-complete orbits appear for background of sufficient length. However, in general this property is undecidable, see [22]. Furthermore, one can construct cellular automata whose Reachability Problem is undecidable on \mathcal{C}_{ap} but whose orbits on backgrounds of any fixed size is always decidable.

4 The Reversible Case

As we have seen, arbitrary cellular automata can have orbits of every c.e. degree. It is thus natural to search for restricted classes of cellular automata which may have less complicated orbits. One natural choice is the class of reversible cellular automata. The degree classification of the reversible cellular automata is indeed somewhat less complicated than for arbitrary cellular automata in the following sense. Consider the *Confluence Problem* for a dynamical system: given two configurations X and Y , is there a configuration Z that is reachable from both X and Y ? In other words, do X and Y lead to the same limit cycle? It is clear that the Confluence Problem, just like Reachability is always c.e. The following result was established in [23].

Theorem 5. *Given any two c.e. degrees \mathbf{d}_1 and \mathbf{d}_2 there is an cellular automaton whose Reachability Problem has degree \mathbf{d}_1 and whose Confluence Problem has degree \mathbf{d}_2 .*

It is clear that no analogous result can hold for reversible systems: X and Y here are confluent only in the trivial case where one configuration is reachable from the other.

The classical reference for reversible computation in the context of the mathematical theory of computation, rather than considerations more closely related to the physics of computation, is Bennett's paper that shows that arbitrary partial recursive functions can be computed reversibly on a suitable Turing machine, see [2]. In the construction, the intended output is copied before the computation is undone using an appropriate history tape. As a consequence, one can compute $\langle x, f(x) \rangle$ reversibly given any partial recursive function f . Somewhat surprisingly, the cost in terms of increased time and space complexity of the computation can be made to be quite modest in Bennett's construction, see [3]. It is also noteworthy that a decade prior to Bennett's paper Lecerf used reversible computation without generating output to establish the undecidability of certain equations, see [10]. Lecerf's construction carries over more naturally to the setting of cellular automata where input/output behavior is problematic. At any rate, for any c.e. set W there is a reversible Turing machine that accepts W .

Reversibility in the context of cellular automata is well-studied, see for example [25] for an overview. In [13, 12] Morita and Harao gave an elegant construction for a reversible one-dimensional cellular automaton that is computationally universal, showing that the Lecerf-Bennett approach can be carried over into the realm of cellular automata. The construction allows one to build one-dimensional reversible cellular automata with relatively little effort. Their argument uses a three-track automaton whose global map is given by the composition of a shearing transformation (the top track moves to the left by one cell while the bottom track moves to the right) followed by the pointwise application of a map $f : \Sigma \rightarrow \Sigma$. More precisely, locally a configuration changes as follows. First the shearing transformation is applied to align x , y and z in one cell. Then (x, y, z) is replaced by $(x', y', z') = f(x, y, z)$.

$$\begin{array}{|c|c|c|c|c|} \hline \cdot & \cdot & x & \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & y & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & z & \cdot & \cdot \\ \hline \end{array} \quad \Longrightarrow \quad \begin{array}{|c|c|c|c|c|} \hline \cdot & \cdot & \cdot & x' & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & y' & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & \cdot & z' & \cdot \\ \hline \end{array}$$

The global map of the cellular automaton is then reversible if, and only if, the local map f is so reversible. In fact, one can even avoid a complete definition of f so long as the defined part does not include any non-injective assignments. At any rate, one has the following result for \mathcal{C}_f , see [24].

Theorem 6. *For every c.e. degree \mathbf{d} the degree class $\mathbb{C}_{\mathbf{d}}$ contains a reversible cellular automaton.*

The construction combines the standard stability trick with reversibility and a suitable simulation by a reversible cellular automaton ρ . The crux of the simulation is again to ensure that unintended configurations do not alter the degree of ρ . Since the cellular automaton is required to be reversible, no simple erasure technique is applicable. Instead, we exploit the upper and lower tracks to carry additional signal bits so that a cell contains three symbols (x_u, y, z_v) where $u, v \in \{0, 1\}$. As long as the local replacements correspond to appropriate actions of the underlying Turing machine the signal bits remain unchanged. If an undesirable event such as the collision of the two tape-heads occurs, the signal bits between the top and bottom-track are interchanged. Initially, in a finite starting configuration all bits in the quiescent part of the top track are 0 and 1 in the bottom track. For orbits that do not correspond to a computation of the Turing machine a signal bit will ultimately escape into the quiescent part and thereby provide a time-stamp for the configuration, which time-stamp renders the whole orbit decidable.

It is not clear whether this approach can be carried over to configuration spaces that lack a “quiescent” part such as recursive configurations: unintended local interactions here could appear in unboundedly many places and there seems to be no obvious way to construct time-stamps as in the finite or almost periodic case.

5 Conclusion

We have seen that intermediate c.e. degrees appear in many places in the study of the computational complexity of cellular automata, albeit in the form of uniform results: all c.e. degrees appear as the complexity of some decision problem or other associated with the automata. As regards the existence of a natural example of a specific intermediate degree the situation is much more difficult though perhaps not entirely hopeless. In a slightly different context H. Friedman has suggested on FOM, a mailing list for the foundations of mathematics, see <http://www.cs.nyu.edu/mailman/listinfo/fom>, that natural intermediate degrees might appear in the form of the theory of a single first-order formula φ in the language $\mathcal{L}(R)$ where R is a single binary relation symbol. Specifically Friedman is interested in the number of quantifiers needed in φ to ensure that the degree of $\text{Th}(\varphi) = \{\psi \in \mathcal{L}(R) \mid \varphi \vdash \psi\}$ is intermediate. The conjecture is that 8 quantifiers might suffice. By the same token, considering sufficiently simple formulae of Peano arithmetic might produce a version of Wolfram’s PCE; to wit, $\text{Th}(\varphi)$ would appear to have degree \emptyset or \emptyset' for all formulae of size no more than 20.

It is unclear how cellular automata and their orbits relate to the notion of a “process” in Wolfram’s PCE. One plausible objection against the use of intermediate degrees as a counterargument to PCE is that the construction relies heavily on information hiding. Indeed, in the standard Friedberg-Muchnik construction of two incomparable c.e. degrees the two sets A and B so obtained have the property of being low, but their disjoint union is complete, see [16]. Thus, if one were to view the construction as a whole as a process then indeed this process would be computationally universal. Orbits of cellular automata would seem to provide little opportunity for information hiding, but the general Reachability problem may not be the right tool to access the information.

If one is willing to adopt different notions of reducibility other lines of inquiry become available. Recent work by Simpson has shown that if one adopts Muchnik degrees as the framework there are natural intermediate degrees. Interestingly, one of these natural examples is based on random reals. One should note that at least one elementary cellular automaton, known as rule 30, exhibits strong pseudo-random behavior and is in fact used as the default random number generator in the computer algebra system Mathematica, see [27]. It is tempting to speculate that the classification of the orbits of rule 30, on sufficiently general types of configurations, might provide another natural source of intermediate behavior. In particular almost periodic configurations might suffice for this purpose.

References

1. J. T. Baldwin and S. Shelah. On the classifiability of cellular automata. *Theoretical Computer Science*, 230(1-2):117–129, 2000.
2. C. H. Bennett. Logical reversibility of computation. *IBM journal of Research and Development*, pages 525–532, 1973.
3. C. H. Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, pages 766–776, 1989.
4. K. Culik and Sheng Yu. Undecidability of CA classification schemes. *Complex Systems*, 2(2):177–190, 1988.
5. M. Davis. *A note on universal Turing machines*, pages 167–175. Princeton University Press, 1956.
6. M. Davis. The definition of universal Turing machines. *Proc. of the American Mathematical Society*, 8:1125–1126, 1957.
7. K. M. Evans. Is Bosco’s rule universal? In *MCU’04*, Sankt Petersburg, 2004.
8. R. M. Friedberg. Two recursively enumerable sets of incomparable degrees of unsolvability. *Proc. Natl. Acad. Sci. USA*, 43:236–238, 1957.
9. L. Harrington and S. Shelah. The undecidability of the recursively enumerable degrees. *Bull. Amer. Math. Soc.*, 6:79–80, 1982.
10. Y. Lecerf. Machine de Turing réversible. Insolubilité récursive en $n \in \mathbb{N}$ de l’équation $u = \theta^n u$, où θ est un “isomorphisme de codes”. *C. R. Acad. Sci. Paris*, 257:2597–2600, 1963.
11. M. Lerman. *Degrees of Unsolvability*. Perspectives in Mathematical Logic. Springer Verlag, 1983.

12. K. Morita. Reversible cellular automata. *J. Information Processing Society of Japan*, 35:315–321, 1994.
13. K. Morita and M. Harao. Computation universality of 1 dimensional reversible (injective) cellular automata. *Transactions Institute of Electronics, Information and Communication Engineers, E*, 72:758–762, 1989.
14. A. A. Muchnik. On the unsolvability of the problem of reducibility in the theory of algorithms. *Dokl. Acad. Nauk SSSR*, 108:194–197, 1956.
15. J. C. Shepherdson. Machine configuration and word problems of given degree of unsolvability. *Z. f. Math. Logik u. Grundlagen d. Mathematik*, 11:149–175, 1965.
16. R. I. Soare. The Friedberg-Muchnik theorem re-examined. *Canad. J. Math.*, 24:1070–1078, 1972.
17. R. I. Soare. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic. Springer Verlag, 1987.
18. K. Sutner. A note on Culik-Yu classes. *Complex Systems*, 3(1):107–115, 1989.
19. K. Sutner. Classifying circular cellular automata. *Physica D*, 45(1–3):386–395, 1990.
20. K. Sutner. De Bruijn graphs and linear cellular automata. *Complex Systems*, 5(1):19–30, 1991.
21. K. Sutner. Cellular automata and intermediate reachability problems. *Fundamentae Informaticae*, 52(1-3):249–256, 2002.
22. K. Sutner. Almost periodic configurations on linear cellular automata. To appear, 2003.
23. K. Sutner. Cellular automata and intermediate degrees. *Theoretical Computer Science*, 296:365–375, 2003.
24. K. Sutner. The complexity of reversible cellular automata. *Theoretical Computer Science*, 325(2):317–328, 2004.
25. T. Toffoli and N. Margolus. Injective cellular automata. *Physica D*, 45(1–3):386–395, 1990.
26. S. Wolfram. Computation theory of cellular automata. *Comm. Math. Physics*, 96(1):15–57, 1984.
27. S. Wolfram. *The Mathematica Book*. Cambridge University Press, 2002.
28. S. Wolfram. *A New Kind of Science*. Wolfram Media, 2002.