Je-LKS
Journal of e-Learning and Knowledge Society
The Italian e-Learning Association Journal
PEER REVIEWED PAPERS
Vol. 14, n.3, 2018
ISSN: 1826-6223 | eISSN: 1971-8829

A GENTLE INTRODUCTION TO COMPUTATIONAL COMPLEXITY THROUGH AN EXAMINATION OF NOODLE MAKING

Luisa Mich[1]
Daniel M. Berry[2]

[1] Department of Industrial Engineering, University of Trento, Trento, Italy
[2] Cheriton School of Computer Science, University of Waterloo Waterloo, Canada

Keywords: Algorithm, Computational complexity, Computational Thinking, Computer Science curriculum, Teaching.

Computational complexity is regarded by many Computer Science students as extremely difficult and as a topic to be avoided. However, the concepts of an algorithm and of computational complexity as a means of characterizing the resource consumption of algorithms are fundamental in Computer Science and are included in all curricula for it. To better motivate students and to increase their interest in computational complexity, this paper suggests introducing it by examining algorithms, a.k.a. recipes, for making noodles. This paper describes several traditional algorithms for making Chinese and Italian noodles and classifies each according to its computational complexity. It compares the power of the algorithms. It considers the nature of variations of the traditional algorithms. It examines machines that implement some of the algorithms. It cites a world speed record for making a large number of noodles using the algorithm with the maximal complexity. It shows how computational thinking and other topics can be introduced in the same

for citations:
Mich L., Berry D.M. (2018), *A Gentle Introduction to Computational Complexity Through an Examination of Noodle Making,* Journal of e-Learning and Knowledge Society, v.14, n.3, 77-92.
ISSN: 1826-6223, e-ISSN:1971-8829 - *DOI: 10.20368/1971-8829/1399*

manner. It concludes by mentioning avenues for further studies.

1 Introduction

A key question in Computer Science is "What can be (efficiently) automated?" (Dennis (Editor) *et. al.*, 1989). Answering this question requires understanding (1) how to devise algorithms to solve problems and (2) how to compare the efficiency of algorithms. Comparing the efficiency of algorithms entails comparing algorithms by their consumption of space and time as they run on the same inputs. A characterization of the run-time space and time consumption of an algorithm as a mathematical function of the size of an input to the algorithm is called the *computational complexity* of the algorithm. Computational complexity is fundamental for computational thinking and therefore for many disciplines, not just for computer science (Wing, 2006; Guzdial, 2008; Yadav, Stephenson & Hong, 2017).

Computational complexity as part of the theory of computability was included in ACM's Curriculum 68 (Atchison *et al.*, 1968). Computational complexity was listed as one of the eight major components of the theory of algorithms and data structures in a framework for the discipline of computing (Denning (Editor) *et al.*, 1989) that served as a basis for a 1989 update to Curriculum 68. The 2001 Joint, ACM and IEEE-CS, Task Force on Computing Curricula included algorithms and complexity as one of the areas, namely Area 1, of the body of knowledge of Computer Science (The Joint Task Force on Computing Curricula, 2001). According to the glossary of the Task Force report, Algorithms and Complexity includes: "Computational solutions (algorithms) to problems; time and space complexity with respect to the relationship between the run time and input and the relationship between memory usage and input as the size of the input grows."

Since the 2001 report, computational complexity has been introduced as a core or secondary topic in courses for undergraduate degree programs in Computer Engineering, Computer Science, Information Systems, Information Technology, and Software Engineering (ACM, undated). An item "Notion of algorithmic complexity" is included even in the ACM's Computer Science Teacher Association K-12 CS Standards (CSTA, 2011).

Beyond the mathematical roots of computational complexity and computational complexity's direct links with other computer science topics such as data structures and programming, computational complexity helps to understand the concept of automation in terms of "what computer technologies can and cannot do, and the impact of individuals, organizations and society of deploying technological solutions and interventions" (Shackelford *et al*., 2006, p. 36).

Computational complexity is notoriously a topic that computer science stu-

dents find difficult (Gasarch, 2015; Trakhtenbrot, 2013; Gasarch, 2017) and, along with general theory of computation, do not like (Meek, 2012; Duan, 2017; Reddit, undated; Wang 2017). It is viewed by many as – shall we say – horribly and impossibly complex! Traditional approaches to teach computational complexity usually treat complexity determination as a mathematical problem, often illustrated by algorithms for searching and sorting of data. To better motivate students and possibly to increase student interest in computational complexity, this paper proposes a novel way to introduce it by describing algorithms, a.k.a. recipes, for making noodles. In particular, Section 2 of this paper compares the computational complexity of several traditional algorithms for making Italian and Chinese noodles. This section also cites a world speed record for making a large number of noodles using a traditional Chinese algorithm that is shown in this paper to require log n time to make n noodles, illustrates machines that implement some of the algorithms, and considers variations of the basic algorithms to make variations of the basic noodles. To demonstrate the applicability and the potential of the proposal, Section 3 reports data on how a previous version of the paper published online in a Computer Science satire conference proceedings[1] (Berry & Mich, 2016) went viral. Section 4 describes some other important Computer Science concepts that can be introduced through examination of noodle making algorithms. The conclusion in Section 5 mentions avenues for further studies. The References section includes an item (Berry & Mich, 2017) that points to an online site which contains (1) the previous version (Berry & Mich, 2016) of this paper and (2) slides for a lecture based on this previous version. Each of these in turn points to sites with videos of the operation of the various algorithms.

2 The Computational Complexity of Chinese and Italian Noodle Making

This section reproduces with a few modifications the essence of the previous version of this paper (Berry & Mich, 2016).

2.1 Introduction to Examination of Noodle Making

Each of the Chinese and the Italians make and eat a large variety of dough-based products of various sizes and shapes. This paper uses "noodle" as general term to name a single unit of any product of this type regardless of its national

[1] The present paper is a variation of this previous version. It includes almost verbatim the sections from the previous paper that describe the noodle making algorithms, their complexity, and a number of open questions. It adds material about the place of computational complexity in Computer Science curricula and the difficulties experienced in teaching it, as motivation for its proposal to consider the previous paper and a talk based on it as educational resources.

origin and regardless of its size and shape[2]. The Chinese call their noodles "miàn tiao" (面条) or just "miàn", and the Italians call their noodles "pasta". Therefore, this paper uses "miàn" and "pasta" when talking about Chinese noodles and Italian noodles, respectively. Note that "miàn" and "pasta" are collective nouns that denote collections of noodles. Thus, this paper needs to use "strand", perhaps prefixed by "miàn" or "pasta" as an adjective, when talking about one unit[3] of miàn or pasta.

This paper presents one key algorithm from each of China and Italy to make the country's most traditional kind of noodles from already made dough of the proper composition for what is being made. Later, it presents some other algorithms, again from China and Italy, for making other kinds of noodles. Each algorithm is characterized by its computational complexity, as a function of the number, n, of noodles produced. There are actually two complexity measures, the *local complexity* and the *global complexity*.

The local complexity is for time required for the algorithm to make one batch of noodles. Generally, the number of noodles that can be made in one batch is limited by a combination of the resources available and the physical properties of the noodle dough. The resource limits that come into play include the amount of flour that can be handled conveniently by the noodle maker, the amount of dough that can be worked on by the noodle-maker's rolling pin, the amount of dough that can be fed at once through a flattening device's rollers, the amount of flattened dough that fits on the noodle-making table, and the amount of flattened dough that can be fed at once through a cutting device. The main physical property of the noodle dough that comes into play is that a noodle with too small a cross section tends to break as it is stretched.

The global complexity is for the time required to make, with successive applications of the algorithm, enough batches to yield all the noodles needed for an occasion. Of course, in a home or in a restaurant that makes noodles to order, usually one batch suffices. In any case, the global complexity is always linear in the number of noodles produced, on the assumption that any algorithm requires about the same amount of time every time it is used to make the same-sized batch of one kind of noodles. Therefore, for each algorithm, only its local complexity is given.

2.2 Traditional Chinese Miàn Algorithm

It appears that the signature variety of miàn in China is the hand-pulled va-

[2] Admittedly, the term "noodle" connotes a string-like product, e.g., spaghetti. Nevertheless, even though many such products are string like, the term is generalized in this paper to include even short products, e.g., maccheroni or macaroni, and even shaped products, e.g., farfalle or bowties.

[3] Just as with "noodle", the term "strand" is used even when the unit is shaped differently from or is shorter than what is normally called a strand or noodle.

riety known as lā miàn, which originated in and around Lan Zhou, the largest city in the Gansu Province of Northwest China. Lā miàn is made by starting with a single strand of dough and repeatedly stretching and folding it to produce a large number of thin strands, the diameter of the final strands depending on the diameter of the single initial strand and the number of folds.

1. The lā miàn maker takes a previously prepared tube of very flexible dough of diameter D and of length L. (L needs to be no longer than the distance across the lā miàn maker's two outstretched arms, and D needs to be no bigger than what the lā miàn maker can grip with one closed hand.) Call this tube of dough "the *initial* bundle".
   - He[4] dusts the bundle with flour.
   - He folds the bundle in half and pinches each end,
     – in one case, to merge two ends into one, and
     – in the other case, to make an end out of a fold.
   - The bundle is now of length $L/2$.
   - By twirling the new bundle like a jump rope, he streches the new bundle back out to the original length, L.
   - The result is a new bundle with twice the number of strands as the previous bundle, and the diameter of each strand in the new bundle is $1/\sqrt{2}$ times the diameter of each strand in the previous bundle. These steps are repeated until the strands are of the desired diameter.

2. The lā miàn maker trims off the ends to leave strands of length $0.9 \times L$. Then, the lā miàn maker lays out the bundle of strands on the table and, in one swift cut perpendicular to the long axis of the strands, cuts all strands to leave two bundles of strands of length $0.45 \times L$.

For a video showing a Chinese chef making lā miàn, see https://www.youtube.com/watch?v=PHoQN9vQwHE, particularly the last minute and a half.

On the assumptions that D is 1 inch, that L is 1 meter[5], and that the final miàn are 1/16 inch (≈ 1.59 mm) in diameter, there are 8 folding and stretching steps, producing 256 trimmed strands each of length 90 centimeters. Then, the final cutting step produces 512 miàn, each of length 45 centimeters.

The local complexity of this traditional Chinese miàn making method is $\log_2 n$ to make $n = 2^m$ miàn in $m - 1$ folding-and-stretching steps making and 1

[4] We use "he" as a singular pronoun to reference a noodle maker of any gender.

[5] The reason that the diameter is in inches while the length and other dimensions are in meters is that it is easier to describe the effect of halving the diameter in terms of binary fractions of an inch.

trimming-and-cutting step[6].

2.3 Traditional Italian Pasta Algorithm

An Italian pasta maker rolls out a ball of the proper dough into a rectangular sheet of the desired thickness *T* and the desired length on one edge, hereinafter called edge *L* (for "length"). An edge that is perpendicular to *L* is called edge *W* (for "width"). (*L* and *W* need to be small enough for an $L \times W$ sheet of dough to be easily worked on by a hand-operated rolling pin). Both sides of the sheet are then thoroughly dusted so that they are not sticky. The sheet is then rolled up very loosely perpendicular to *L* so that the resulting tube is of length equal to that of *W*. The pasta maker decides the type of pasta that is being made to determine the width *w* of one strand. Ideally, the width *W* of the sheet is divisible integrally, *n* times, by *w*. For fettuccine, *w* is smaller than for lasagne.

1. The pasta maker uses a knife to cut away a section of the tube of width *w*.
2. The pasta maker unrolls the section into a strand of width *w* and of length equal to that of *L*.

This cutting and unrolling of sections is performed $n - 1$ times and then the remaining section is unrolled to produce the last strand of a total of *n* strands. All of this cutting and unrolling must be done quickly to prevent the rolled up tube from sticking to itself.

So, if one is making 30-centimeter long fettuccine whose cross section is 1/4 by 1/16 inch, then, *w* must be 1/4 inch, *T* must be 1/16 inch, *L* must be 30 centimeters, and *W* can be anything that be is less then the length of pasta maker's rolling pin and is a multiple of *w*. Let's assume that *W* is 8 inches. Then from one 8 inch by 30 centimeter sheet, the pasta maker will need $8 \times 4 - 1 = 31$ cuts to make 32 strands. For wider pasta, such as lasagne, fewer cuts are needed.

The local complexity of the algorithm is linear in the number of strands, *n*, made from $n - 1$ cuts and *n* unrollings in one sheet of dough.

There are at least two devices that allow cutting a prepared sheet of dough into a lot of strands in one step:

- a pasta cutter rolling pin whose cutting ribs are spaced *w* apart and
- a pasta making machine whose cutting blades are spaced *w* apart.

With either of these devices, there is no need to roll up the sheet and cut away one strand at a time. Instead,

- the cutting rolling pin is rolled once over the flat sheet of prepared dough, leaving the strands flat on the table with no need to unroll, or
- the sheet is fed through the machine, and the strands come out of the

[6] We are assuming that trimming and cutting take about the same amount of time asdoes folding and stretching.

machine already unrolled.

Several cutter rolling pins with ribs spaced different distances apart can be seen at the Eppicotispai Group's Website (Eppicotispai Group, undated).

The algorithm embodied by each of these devices can be described as a parallel, vector processing algorithm. Thus, the local complexity of the algorithm to make *n* strands from one prepared sheet with either of these devices is constant. That is, all *n* strands are made at the same time, in the time required to roll the cutting pin over the sheet or to feed the sheet though the machine.

2.4 World Record Setting Chinese Lā Miàn Maker

In 1993, Kin Jing Mark, who holds the Guinness World Record as the fastest human noodle maker, established a world record stretching out 4,096 lā miàn hand-pulled dragon beard noodles in 41.34 seconds (Youtube, 2007). The number, 4096, of strands that Kin Jing Mark made, is telling: $4096 = 2^{12}$. It is clear that no one spent time counting the individual strands to arrive at 4096. It is equally clear that the number of folds was counted and that number was used as the exponent of 2 to calculate the number of strands. Thus on average, Kin Jing Mark did one fold and stretch every 3.445 seconds. Clearly, the cook and the people who made the video understand the exponential growth of the number of strands in the algorithm.

2.5 Automation of Algorithms

Searching for "Italian pasta making machines" on the Web finds pasta sheeters and cutters that automate the Italian pasta-making process. These machines simulate the human pasta maker's behavior, to make so-called perfectly formed pasta every time.

There does not appear to be any machines that automate the making of Chinese lā miàn. There are machines that automate the mixing and kneading of the dough, but there do not appear to be any machines that automate the folding and stretching. Perhaps the main reason that lā miàn are called in English "hand-pulled" is that they *must* be made by a human's hand.

Indeed, one day, an engineer from Barilla (one of Italy's largest past manufacturers) and a Japanese visitor came to observe Paola Abraini making Italian pasta a method that is very similar to the traditional Chinese miàn method (Locci, 2016), with the aim of building a machine implementing the method. They left after concluding that such a machine was impossible[7] (Pinna, 2016).

[7] Un giorno è venuto un ingegnere della Barilla con un giapponese; hanno osservato, volevano costruire una macchina e produrlo. Sono andati via sconsolati: impossibile.

2.6 Other Methods of Making Noodles

China does have other methods of making miàn (LinLiu, 2013; Wikipedia, undated*a*):

- Cut (qie): A sheet of dough is cut into strands of the desired width, as in the traditional Italian pasta algorithm of Section 3. The local complexity of this process is linear in the number of strands produced.
- Squared (piàn): As one is making cut miàn, directly above an open pot of boiling water, each (long) strand is torn by hand into square-sized pieces (short strands). The local complexity of this process is linear in the number, n, of pieces produced: If s long strands are produced with $s - 1$ cuts, and from each long strand are produced p short strands with $p - 1$ tears, then $s \times p = n$. The total number of steps is $(s - 1) + (s - 1) \times (p - 1) = (s - 1) \times p$, which is approximately $s \times p = n$.
- Extruded (jíya): Dough is pushed through a die with holes of the desired shape to form strands, one per hole in the die. The local complexity of this process is constant, since all the strands are produced at the same time.
- Kneaded (róu): A small ball of dough is worked on a flat surface to form it into a strand of the desired shape. The local complexity of this process is linear in the number of strands produced.

Also Italy has other methods of making pasta (Hildebrand & Kenedy, 2010, Wikipedia undated*b*):

- Short cut: As one is following the traditional Italian pasta algorithm of Section 3, the s unrolled (long) strands are laid out in parallel, side-by-side into a striped sheet. Then, $p - 1$ equally spaced cuts perpendicular to the axis of the length of the strands are applied across the whole sheet, to produce $s \times p = n$ rectangular pieces (short strands). The local complexity of this process is in the order of the square root of the number of pieces produced: The complexity analysis for this process starts as for the production of $n = s \times p$ squared piàn. The difference is that all s long strands are cut together in only $p - 1$ cuts. Thus, the total number of cuts is $s - 1 + p - 1$, which is approximately $s + p$. If the sheet is close to being a square, then $s \approx p$, and $s + p \approx 2\sqrt{n}$, since $s \times p = n$. In a complexity estimate, a costant multiplier of $\sqrt{n}$ can be ignored, because the main contributor to the growth of $2\sqrt{n}$ is $\sqrt{n}$, and not the constant multiplier. There is a variation of the pasta cutter rolling pin, mentioned in Section 3, that has cutting ribs running along the long axis of the pin, perpendi-

cular to the strand-cutting ribs. This variation is for producing a whole sheet's worth of short-cut pasta in one roll of the pin over a prepared sheet of dough. The local complexity of this method of making short-cut pasta is constant.

- Extrusion: Dough is pushed through a die with holes of the desired shape to form strands, one per hole in the die. The local complexity of this process is constant, since all the strands are produced at the same time.
- Short-cut extrusion: Each extruded pasta long strand is cut perpendicular to the length of the strand into short pieces. The local complexity of this process is in the order of the square root of the number of pieces produced, by the same analysis as for the above short-cut pasta.

Additional shaping may be applied to the pasta produced by any of the described methods.

There are machines that automate all the various ways of making Italian pasta, for example, as shown under the "Products" menu at the Arcobaleno Website Arcobaleno, LLC (undated).

2.7 Comparison of Algorithms

The Chinese repeated-folding-and-stretching algorithm, with its logarithmic complexity is significantly more powerful than any Italian cutting-based algorithm, with linear complexity, in two different ways:

1. The logarithmic-complexity algorithm can generate so many more noodles in a time duration than can any linear-complexity algorithm, particularly when the noodle maker is folding and stretching quickly, and he goes beyond six folds. Twelve folds and stretches in 41.34 seconds suffices to make 4096 noodles. Making 4096 noodles by any linear-complexity algorithm would require a *lot* more than 41.34 seconds.
2. When the two algorithms are operated totally manually, it is a lot easier to achieve uniformity in the cross section of the noodles with the repeated-folding-and-stretching algorithm than with any cutting-based algorithm.

2.8 Variations of the Basic Noodles

The different algorithms for making noodles lead to variations of the basic noodles that we see in the two countries. The fact that a flat sheet of dough is cut into strands that become the noodles suggests cutting the sheet into other shapes. Hence, we see noodles in the shapes of triangles, squares, rectangles, circles, stars, etc. Once we have these different shapes, we begin to see yet

other variations, such as pinching a rectangle into a bowtie, molding a circle into a shell. Once we have shaping and pinching, with the addition of a bit more water, pinching can be used to paste edges together. Then rolling and pasting a wet rectangle yields a tube that can be filled. Covering part of a shape with some filling and folding and pinching wet edges around the filling yields filled tortellini. Once all this is automated, shapes that can be made by machinery become possible.

The Chinese lā miàn algorithm does not lend itself to these cutting-based variations. There are variations in the length and diameter of the noodles, the raw material used to make the noodles, and the twistiness of the noodle achieved by variations in the process of drying the noodles, e.g., by spiraling the wet noodles around a dowel of an appropriate diameter.

2.9 Conclusion to Examination of Algorithms for Noodle

Making There are a number of questions, each of which can trigger a discussion about other computational concepts and which is useful to better understand the roles of algorithms and their automation.

- Perhaps, the space required for an algorithm should be considered. Is there a meaningful space–time tradeoff? Does the size of the available kitchen make a difference, e.g., as for a home versus a restaurant kitchen?
- What is the interaction between the algorithms and the ingredients used to make the dough?
- What is the interaction between the algorithms and the issue of fresh versus dried noodles?
- What is the interaction between the algorithms and the local culture?
- Can a Chinese algorithm be applied in Italy and can an Italian algorithm be applied in China? As a matter of fact, One Italian pasta maker, Paola Abraini, uses a method that is very similar to the traditional Chinese miàn method to make Italian pasta (Pinna,2016).
- How easily learned is each algorithm?
- How automatable is each algorithm?
- Which algorithm is most appropriate to use in a restaurant in which food is made to order?
- What are the empirically determined threshold values for n (the number of noodles made in a batch), below which the traditional basic and parallel Italian algorithms are more efficient locally than the traditional Chinese algorithm?

3 Evidence of the Effectiveness of the Gentle Introduction

This section offers indirect evidence that the gentle introduction appeals people.

Author Mich uploaded to ResearchGate (2016) the previous version of the paper, which had been published in the proceedings of SIGBOVIK, a Computer Science satire conference (Berry & Mich, 2016). ResearchGate reports that the paper went viral on September 2016, and that as of 10 October 2018, the paper had 1888 reads. There are an unknown number of reads of the copy at Author Berry's Web site (Berry & Mich, 2017).

One version or another has been mentioned in a variety of other sites. For example, one continues the tongue-in-cheek application (Imgur, undated) of complexity to a serious recipe (Rebrn, undated). This tongue-in-cheek application is shared by some other sites, including that of Reddit Programmer Humor (Reddit Programmer Humor, undated; Sizzle, undated; Imgur, 2016). In another, Dan Eastwood cites the ResearchGate copy in a brief blog titled "Noodling around, FOR SCIENCE!" (Eastwood, 2016).

A Google search in Safari on a Mac for the exact title of the previous version achieves about 104 hits as of 10 October 2018. Most of the hits are pointers to pages that mention that previous version.

The lecture Berry gave at the publishing venue, the slides of which can be found at the Author Berry's Web site (Berry & Mich, 2017), was well received. He has given the lecture four more times as part of his department's outreach to high school students to convince them to apply to study Computer Science in his department. The lecture was well enough received that he was asked to plan on giving it again the next year.

More recently, DocPlayer, an educational resource, began offering at its Website a paged copy of the paper and the possibility of downloading the full PDF for free (DocPlayer, undated*a*). The home page of the organization says (DocPlayer undated*b*),

> What tasks does our website help to tackle? Our website makes it easy for you to find books that will help prepare for an examination, complete reports and research papers, as well as self-study books in various areas. Educational Library of the web resource contains thousands of training manuals, articles and books in a wide variety of academic subjects.

The authors did not ask DocPlayer to put it there. Someone else, perhaps at DocPlayer, decided that it was a good resource for DocPlayer's site. If that someone was from outside DocPlayer, the both that someone and DocPlayer agreed!

4 Other Topics and Concepts Introduceable Through Examination of Noodle Making

This section shows how a number of core topics and concepts of Computer Science could be gently introduced through examination of noodle making. Only a few are given here, but there are possibly others.

The most important concept in Computer Science is that of an algorithm. According to the everyday definition, "algorithm" is synonymous to "procedure". However, a precise definition requires that to be an algorithm, a procedure has to satisfy five proprieties, namely, it has to (1) be finite, (2) be defined (unambiguous), (2) have input, (2) have output, and (5) be effective (Knuth, 1997, pp. 4–5). Each of these features could be introduced through examination of noodle making.

Knuth also compares the concept of algorithm with that of a cookbook recipe, emphasizing that a "recipe presumably has the qualities of finiteness (although it is said that a watched pot never boils), input (eggs, flour, etc.), and output (TV dinner, etc.), but it notoriously lacks definiteness". To be definite, the actions to be carried out in each step must be rigorously and unambiguously specified. That is why formal languages have to be used when an algorithm cannot be described using mathematical formula. As for effectiveness, following a noodle making procedure – even with a video available illustrating the procedure – is still quite difficult, in both the Italian and the Chinese ways. Showing this point could be useful to introduce the concept of executor, i.e. the processor, that must have some specific characteristics to be able to accomplish the procedure's steps. In the case of noodle making, the executor, i.e. the cook, must have enough expertise in following imprecise recipes.

Any algorithm has to be described in terms of input and output. The actual output and its quality depends on the input, a.k.a. "garbage in, garbage out". If you do not use the right kind of flour, i.e., grano duro for Italian pasta, your noodles will not be able to be be cooked al dente, but even worse, in both the Italian and the Chinese procedures, if the dough is not correct, e.g., if it is sticky, you are not able to produce noodles; you will make mush.

Finally, for a given problem there could be different algorithms, and their analysis in terms of complexity is useful to adopt the most adequate according to the input, output, and available processors. For example, if you lack expertise, you can buy a machine to make noodles.

In addition, some noodle-making tasks can be accomplished in parallel, and others cannot. Thus, the concept of parallel processing can be introduced.

Another important point in Computer Science is computational thinking (Wing, 2006). Computational thinking is the ability to think of real-life problems, not involving computing, in an algorithmic way, i.e., to understand

everyday phenomena as instances of computations, driven by algorithms performed by some agent, not necessarily a computer, e.g., human beings. Understanding the whole process of noodle making as a computation and seeing the various methods of making noodles as algorithms are manifestations of computational thinking. Indeed, Author Berry came to this realization when in a fancy Hong Kong restaurant in 1987, the cook prepared his noodles to order, in front of him at the table, using the traditional Chinese miàn algorithm. He remarked to his meal companions, most of whom were computer scientists, that this cook was using a log-base-2 algorithm!

Even deeper computational thinking is demonstrated when cooks begin to understand the computational power of a recipe in which each step doubles the number of items produced. For example, Paola Abraini states that she produces two, four, eight, sixteen, thirty-two strands, always thinner, up to 256 strands (Pinna, 2016). She understood that 256 is probably the limit because more than that, the strands might be so thin that they will break. Abraini trusts that the number of strands doubles with each fold and stretch and apparently feels no need to count the strands to check. The same trusting occurs in reporting that the World's record setting production of Chinese miàn produced 4096 noodles. That the number reported is exactly a power of two indicates that the reporter counted the number of folds, 12, to determine that the final number of noodles is $2^{12} = 4096$ and not the number of noodles. If he or she had attempted to count the noodles, it is unlikely that the count would have come out at exactly 4096 (from miscounting, not from there being a different number of noodles).

Conclusion

This paper illustrates a new and original approach to introduce the teaching of computational complexity, a topic of great relevance to the concept of algorithms. The idea is to observe, understand, analyze, and compare several Italian and Chinese ways of making noodles. The differences between constant, linear, and logarithmic algorithms are brought to life by examining the algorithms embodied in the real-life activity of working with dough to make noodles.

The approach can be considered both a gentle and an unplugged (CS Unplugged, undated) introduction, as it mitigates the students' traditional aversion to the subject combined with the traditional (boring) catalog of sorting and search algorithms by appeal to the ever popular activity of food preparation. Students' curiosity can be stimulated also by the cultural context and the discovery of how the world's record was achieved by use of the power of a logarithmic algorithm. The paper also describes other computational concepts that could be introduced through the discussion of different ways of making noodles.

At a more general level, the approach offers the possibility to involve teachers of subjects other than computer science, for example, home economics, geography, or history, in a multidisciplinary teaching approach, which is often recommended for teaching in secondary schools.

The paper offers testimonials from colleagues and online data about the appeal of this approach to introducing computational complexity.

Future work needs to investigate the efficacy of the approach at different school levels, in different curricula, and extensions of the approach for teaching other computer-science concepts, as suggested in Section 4.

Through the URL in an item (Berry & Mich, 2017) in the References section, additional materials are provided at a site containing the previous version of this paper (Berry & Mich, 2016) and slides for a lecture based on this previous version. Each of these documents contains URLs pointing to videos showing the algorithms being executed by cooks.

Acknowledgments

Thanks to Nachum Dershowitz, Dick Kemmerer, Mickey Krieger, Chryss Mylopoulos, Frank Tompa, and Dave Tompkins for comments on an earlier draft. Frank, in particular, offered some insights that led to a number of improvements to the paper.

REFERENCES

ACM (undated), *Curricula recommendations*. http://www.acm.org/ education/curricula-recommendations, (accessed on 10 October 2018).

Arcobaleno, LLC (undated), *Homepage*. https://arcobalenollc.com, (accessed on 10 October 2018).

Atchison, W. F., Conte, S. D., Hamblen, J. W., Hull, T. E., Keenan, T. A., Kehl, W. B., McCluskey, E. J., Navarro, S. O., Rheinboldt, W. C., Schweppe, E. J., Viavant, W. & Young, Jr., D. M. (1968), Curriculum 68: Recommendations for academic programs in Computer Science: A report of the ACM Curriculum Committee on Computer Science, *Communications of the ACM* 11(3), 151–197.

Berry, D. M. & Mich, L. (2016), The computational complexity of Chinese and Italian noodle making, in *Proceedings of SIGBOVIK 2016*, pp. Paper 6, pp. 0x248fd1aa778f4c65964afe5512748fe503a3d796ec35721e3fb68c0a7063dff5, ff. http://sigbovik.org/2016/proceedings.pdf.

Berry, D. M. & Mich, L. (2017), *Pasta paper materials*. https://cs.uwaterloo.ca/~dberry/FTP_SITE/tech.reports/PastaPaper/, (accessed on 10 October 2018).

CS Unplugged (undated), *Computer science without a computer*. http://csunplugged.org, (accessed on 10 October 2018).

CSTA (2011), *CS Standards Crosswalk with CSTA K-12 Computer Science Standards for State/District/Course Standards*. http://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/Docs/Curriculum/Crosswalk_Templates/SampleCrosswalkGrades6-12CST.pdf?hhSearchTerms=%22complexity%22, (accessed on 10 October 2018).

Denning (Editor), P. J., Comer, D. E., Gries, D., Mulder, M. C., Tucker, A., Turner, A. J. & Young, P. R. (1989), Computing as a discipline, *Communications of the ACM* 32(1), 9–23.

DocPlayer (undated*a*), *The computational complexity of Chinese and Italian noodle making*. https://docplayer.net/ 63565057-The-computational-complexity-of-chinese-and-italian-noodle-making. html, (accessed on 10 October 2018).

DocPlayer (undated*b*), *We offer you effective and free publishing and information sharing tools*. https://docplayer.net, (accessed on 10 October 2018).

Duan, J. (2015), *Sorting is boring: Computer Science education needs to join the real world*. http://www.huffingtonpost.com/jessie-duan/sorting-is-boring-compute_b_6675650.html, (accessed on 10 October 2018).

Eastwood, D. (2016), *Noodling around, FOR SCIENCE!*. https://plus.google.com/+DanEastwood/posts/Mkg3fQbamzZ, (accessed on 10 October 2018).

Eppicotispai Group (undated), *LET'S MAKE PASTA TOGETHER: Spaghetti, tagliatelle, fettuccine, pappardelle*. http://www.eppicotispai.it/?we-love-pasta/spaghetti-tagliatelle-fettuccine-pappardelle, (accessed on 10 October 2018).

Gasarch, B. (2015), *Why do students do this?*. http//blog.computationalcomplexity.org/2015/01/why-do-students-do-this.html, (accessed on 10 October 2018).

Gasarch, B. (2017), *Students try to memorize rather than understand! Who knew! (Everyone)*. http://blog.computationalcomplexity.org/2017/05/ students-try-to-memorize-rather-than.html, (accessed on 10 Oc- tober 2018).

Guzdial, M. (2008), Education: Paving the way for computational thinking, *Communications of the ACM* 51(8), 25–27.

Hildebrand, C. & Kenedy, J. (2010), *The Geometry of Pasta*, Box Tree, London, UK.

Imgur (2016), *A sharing of Imgur (undated)*. http://imgur.com/nrfWekQ, (accessed on 10 October 2018).

Imgur (undated), *A commenting on Rebrn (undated)*. http://i.imgur.com/nrfWekQ.png, (accessed on 10 October 2018).

Knuth, D. E. (1997), *The Art of Computer Programming*, Volume 1 (3rd Ed.): Fundamental Algorithms, Addison Wesley Longman, Redwood City, CA, USA.

Lin-Liu, J. (2013), *On the Noodle Road: From Beijing to Rome, with Love and Pasta*, Riverhead Books, The Penguin Group, New York, NY, USA.

Locci, M. (2016), *I segreti dei "fili di dio", l'arte di Paola Abraini a Nuoro*, Corriere della Sera TV. http://video.corriere.it/i-segreti-fili-dio/ 129ebe82-9b99-11e6-92af-45665cb81731.

Meek, T. (2012), *Math nerds vs. code monkeys: Should Computer Science classes be more practical?*. https://blog.smartbear.com/careers/ math-nerds-vs-code-monkeys-should-computer-science-classes-be-more-practical, (accessed on 10 October 2018).

Pinna, A. (2016), *Chi `e la custode dei "fili di dio" la pasta che nessuno sa imitare*, Corriere della Sera. http: //www.corriere.it/cronache/16_ottobre_26/chi-custode-fili-dio-d0fb0a94-9bc5-11e6-92af-45665cb81731.shtml?refresh_ce-cp.

Rebrn (undated), *[oc] Made ramen the other day. hardest thing I've ever attempted but was absolutely worth it*. http://rebrn.com/re/ oc-made-ramen-the-other-day-hardest-thing-ive-ever-attempted-but-2815792/, (accessed on 10 October 2018).

Reddit (undated), *Why do many comp sci majors hate math?*. https://www.reddit.com/r/compsci/comments/42fo42/why_do_many_comp_sci_majors_hate_math, (accessed on 10 October 2018).

Reddit Programmer Humor (undated), *A sharing of and a commenting on Imgur (undated)*. https://www.reddit.com/r/ProgrammerHumor/comments/5270w2/programmer_redditor_explains_why_ramen_is, (accessed on 10 October 2018).

ResearchGate (2016), *The computational complexity of Chinese and Italian noodle making*. https://www.researchgate.net/publication/299605105_The_Computational_Complexity_of_Chinese_and_ Italian_Noodle_Making, (accessed on 10 October 2018).

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., Cross, J., Impagliazzo, J., LeBlanc, R. & Lunt, B. (2006), Computing curricula 2005: The overview report, *SIGCSE Bulletin* 38(1), 456–457.

Sizzle (undated), *A sharing of Imgur (undated) and Reddit (undated)*. https://onsizzle.com/i/oc-made-ramen-the-other-day-hardest-thing-ive-ever-2336720, (accessed on 10 October 2018).

The Joint Task Force on Computing Curricula (2001), Computing curricula 2001, *Journal on Educational Resources in Computing* 1(3es), Article 1.

Trakhtenbrot, M. (2013), Students misconceptions in analysis of algorithmic and computational complexity of problems, in *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE)*, pp. 353–354.

Wang, D. (2017), *Why do so few people major in computer science?*. http://danwang.co/why-so-few-computer-science-majors, (accessed on 10 October 2018).

Wikipedia (undated*a*), *Chinese noodles— Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Chinese_noodles, (accessed on 10 October 2018).

Wikipedia (undated*b*), *List of pasta— Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/List_of_pasta, (accessed on 10 October 2018).

Wing, J. M. (2006), Computational thinking, *Communications of the ACM* 49(3), 33–35.

Yadav, A., Stephenson, C. & Hong, H. (2017), Computational thinking for teacher education, *Communications of the ACM* 60(4), 55–62.

YouTube (2007), *'The ring of truth: Atoms' featuring Chef Kin Jing Mark*. https://www.youtube.com/watch?feature=player_embedded&v= auhHl5-6VdY, (accessed on 10 October 2018).