

# rebuttal FST TCS

---

date: 30 August 2021 from: [fsttcs2021@easychair.org](mailto:fsttcs2021@easychair.org)

Thank you for your submission to FSTTCS 2021. The FSTTCS 2021 rebuttal period is now open, and it will close at 23:59, September 1, AoE.

During this time, you will have access to the current state of your reviews and have the opportunity to submit a response. Please keep in mind the following during this process:

- The response should mainly focus on any factual errors in the reviews and any questions posed by the reviewers. It must not provide new research results or reformulate the presentation. There is no limit on the length of the response, but please try to be concise and to the point.
- The rebuttal period is an opportunity to react to the reviews, but not a requirement to do so. Thus, if you feel the reviews are accurate and the reviewers have not asked any questions, then you do not have to respond.
- The reviews are as submitted by the PC members, without any coordination between them. Thus, there may be inconsistencies. Furthermore, these are not the final versions of the reviews. The reviews can later be updated to take into account the discussions at the program committee meeting, and we may find it necessary to solicit other outside reviews after the rebuttal period.
- The program committee will read your responses carefully and take this information into account during the discussions. On the other hand, the program committee will not directly respond to your responses, either before the program committee meeting or in the final versions of the reviews.
- Your response will be seen by all PC members who have access to the discussion of your paper, so please try to be polite and constructive.

The reviews on your paper are attached to this letter. To submit your response you should log on the EasyChair Web page for FSTTCS 2021 and select your submission on the menu.

Best wishes, Chandra (Track A) and Mikolaj (Track B)

---

## REVIEW 1

SUBMISSION: 89 TITLE: Symbolic Weighted Language Models and Quantitative Parsing over Infinite Alphabets AUTHORS: Florent Jacquemard, Philippe Rigaux and Lydia Rodriguez de la Nava

----- Overall evaluation ----- The paper introduces symbolic weighted automata (swA) as a generalization of finite automata in two directions: first, like symbolic automata, they allow infinite input alphabets and second, like weighted automata, they map each input word to a value in a semiring. In the same spirit, symbolic weighted transducers (swT) and symbolic visibly pushdown automata (sw-VPA) are introduced.

Based on these concepts, the paper introduces a variety of constructions. The first one is a Bar-Hillel Perles Shamir construction for a given swT and a given swA. The second one performs the closure of two sw-VPA under the semiring operations. The third one computes, for a given sw-VPA, the best word with respect to the natural order of the underlying semiring.

Finally, the paper introduces the framework of Symbolic Weighted Parsing, which, given

- a word from some input alphabet,
- an sw-VPA over some target alphabet, and
- an swT mapping words from the input alphabet to the internal symbols of the sw-VPA's alphabet, aims to compute a nested word which minimizes both the weight assigned to it by the sw-VPA and the distance to the input word (which is computed by the swT). The authors show that this problem can be solved in polynomial time by combining the three constructions introduced earlier. Moreover, they motivate their work by the example of the automatic transcription of music recordings into music scores.

The formalism of symbolic weighted automata is not original to the paper. It was already introduced by Herrmann and Vogler (2016) and later revised by Herrmann (2020) under the name of weighted symbolic automata with data storage. This literature subsumes swA (by choosing the trivial storage), swT (since transducers can be understood as particular weighted automata), and sw-VPA, for which the data storage VP was defined. Moreover, the earlier work is more general, since it allows unital valuation monoids (rather than just semirings) as weight structures. They authors of the present paper should refer to the earlier publications and clearly distinguish between adopted concepts and their own work.

Moreover, the authors claim that their framework is a generalization of previous approaches to weighted parsing, in the sense that it allows infinite input alphabets. However, some of the previous approaches allowed more general weight structures than just semirings, so their are incomparable to the present paper.

The paper is very well structured and it was a pleasure to see all three automaton models and constructions fit together in the final section.

However, the original content seems to have been squeezed into the final pages of the paper, while in the beginning and in the center many pages were used to repeat the concepts of Herrmann and Vogler (2016).

Moreover, the appendix contains interesting remarks regarding the relation between nested words and parse trees. There are several occasions where the paper refers to the appendix, which should be avoided, since the appendix is not a part of the published conference paper.

I enjoyed the running example of automatic music transcription which serves to illustrate every introduced concept and brings a different domain into consideration. The overall impression of the paper is spoiled by many careless mistakes, such as formulas sticking out into the page margin (see below for a list).

Overall, I think that the paper cannot be published in its present form because the introduction of symbolic weighted automata without reference to the original literature could be seen as plagiarism. I think, however, that the original work presented in the paper can be of interest to the scientific community. This is why I opt to weakly reject the paper. In order for the paper to be published, its focus should be shifted from the theory of symbolic weighted automata to the framework of symbolic weighted parsing.

In particular, the authors could spend fewer pages repeating already established concepts from the literature and integrate Appendix D into the main paper.

REMAINS TO DO : references :

- ☐ 75: maybe cite Parsing as intersection by Nederhof and Satta (2003)

- ☐ 126: you should cite Knuth's generalization of Dijkstra's algorithm for superiority, he defines the notion for functions of arbitrary arity, so not just semirings (but multioperator monoids); you should give an example for superiority in the semiring notation (with  $\otimes$ ) Moreover, the 'corresponds' is wrong, since superiority generalizes non-negative edge weights
  - ☐ PS: Knuth's definition of superiority includes monotonicity
  - ☐ 143: Lemma 4 is Lemma 3 in [21]
  - ☐ 195: references: there is earlier research on symbolic automata, e.g. Veanes (2013)
  - ☐ Ref [22] and [23] both seem to refer to the same work by Mohri
- 
- ☐ 51--53: the jump from automata to grammars is uneasy and not really necessary, as the paper mainly deals with automata
  - ☐ 57: weighted automata don't have ASTs, but runs
  - ☐ 60: other approaches use multioperator monoids or valuation monoids, so 'at minima a semiring' is wrong
  - ☐ 116:  $\oplus$  'combines two values into a single one' as well (in fact, every binary operation does)
  - ☐ 136: what implies?
  - ☐ 153: quantify  $x$  and the family  $(y_i)$  for  $i \in \mathbb{N}$
  - ☐ 155--156: alphabets are usually finite and non-empty sets
  - ☐ 159: denoted by; also it seems like  $\bar{\Phi}$  is a  $(\Sigma \cup \Sigma \times \Delta)$ -indexed family, why don't you introduce it like this
  - ☐ 161--171: use an itemize environment
  - ☐ 162--165: I think these should also hold for  $\Delta$ , why don't you write this explicitly?
  - ☐ 188: the definition of  $\bigoplus_{\Delta}$  can be seen to be implicit. Please fully quantify the alphabets of a label theory
  - ☐ 208: What does  $T$  compute?
  - ☐ 219: resp.\@
  - ☐ 220--221: you mean that a term CONTAINING  $w_{ij}(\dots)$  equal to 0 is ignored
  - ☐ 238: text in the margin, also it is not clear whether the if refers to the entire line or just the right definition
  - ☐ 284:  $x$  is a variable, so it should be typeset in math mode
  - ☐ 291: I'd at least say once that  $\Delta_i$ ,  $\Delta_c$ , and  $\Delta_r$  are pairwise disjoint (or explain the symbol)
  - ☐ 297: the extension here is not as in Section 3, there, the different function hat at least similar types
  - ☐ 299: every line expands into the margin; moreover, in the rightmost equation of line three, in the application of  $w_c$ ,  $p'$  and  $q'$  are swapped
  - ☐ 316: equation extends into margin

- ☐ 331: I think the application of  $w_c$  is wrong (same with  $w_r$  and  $w_i$  later), shouldn't the : be a comma and what is  $q$  in the middle?
- ☐ 336: how are states (which are in  $Q$ ) pushed onto the stack if  $\Gamma = \Delta_c \times P$ ?
- ☐ 339: what are  $j$  and  $c$ , do you mean  $i$  and  $d$ ?
- ☐ 341: shouldn't there be a transition for returning from a measure?
- ☐ 348: for? do you mean accepted by? but the language of an swA isn't defined either
- ☐ 354: readers may have an easier time seeing this argument if you say that  $S$  is extremal. You should say in the preliminaries that  $S$  is extremal if and only if  $\leq_\oplus$  is total.
- ☐ 371: How is it ensured that the symbol  $< c, p >$  isn't popped and pushed again later during the computation?
- ☐ 376: I do not understand the and at termination, and its correctness part. What do you want to express?
- ☐ 379: Almost all update equations need to compute infinite sums. Don't you think that the assumption of  $\bar{\Phi}$  being effective is pretty strong and gives you the result 'for free'? I think you should at least elaborate on this and say to what degree your effective label theory is a generalization of the decidable label theory of unweighted symbolic automata.
- ☐ 629: typical references for regular tree grammars are Brainerd (1969) and Gecseg and Steinby (1984)
- ☐ 654: an intuition for the  $\bar{q}$  would have made understanding the construction far easier

R3 :

- ☐ l. 100. Shouldn't the first measure end before the second starts? This would make  $O$  a forest rather than a tree - easily fixed with a "piece" root element.
- ☐ l. 220. The sentence starting here is somewhat confusing (and possibly wrong, depending on if  $q'$  and  $q''$  are intended to be distinct). I suppose that the point is that one can quit the recursion "early" whenever  $w_{ij}$  returns 0, but the language could be clearer, something like "Since 0 is absorbing for  $\times$ , and the identity element for  $+$  in  $S$ , if  $w_{ij}(q, a, b, q'')$  is 0 (meaning there is no transition from  $q$  to  $q'$  reading  $a$  and  $b$ ) that entire term can be ignored in the sum."
- ☐ l. 288: same as on line 100, measure 1 should maybe end before measure 2
- ☐ l. 434:  $swM??$

List of mistakes:


- ☒ 17: who -> which; missing comma after swT
- ☒ 18: superfluous comma
- ☒ 26: compilers or interpreters for programming languages
- ☒ 27: superfluous comma
- ☒ 32: superfluous comma
- ☒ 33: Western
- ☒ 43: skip commas around parentheses; 'with' is not an optimal conjunction
- ☒ Figure 1: do not start sentences with symbols; add an 'and' before  $S$
- ☒ 47: Boolean Formulas

- ☐ 51--53: the jump from automata to grammars is uneasy and not really necessary, as the paper mainly deals with automata
- ☐ 57: weighted automata don't have ASTs, but runs
- ☐ 60: other approaches use multioperator monoids or valuation monoids, so 'at minima a semiring' is wrong
- ☒ 70: the second comma is superfluous
- ☒ 74: missing period and space after  $\oplus$
- ☐ 75: maybe cite Parsing as intersection by Nederhof and Satta (2003)
- ☒ 77: missing space after period
- ☒ 84: convenient in this context -> for convenience?
- ☒ 87: superfluous comma
- ☐ 116:  $\oplus$  'combines two values into a single one' as well (in fact, every binary operation does)
- ☒ 118--119: I would say:  $\dots \leq_{\oplus}$  which is called the natural ordering of  $S$  and is, for every  $\dots$ , defined by ...
- ☒ Also, you should be consistent with using 'for all' or 'for every' in quantifications (I would prefer for every)
- ☒ 122: if it IS; I'd also add a comma after total
- ☒ 123: Lemma 2 is Lemma 2 in [21], you should refer to it
- ☒ 125: you write 'then', but the following sentence seems to be disconnected from the previous lemma
- ☐ 126: you should cite Knuth's generalization of Dijkstra's algorithm for superiority, he defines the notion for functions of arbitrary arity, so not just semirings (but multioperator monoids); you should give an example for superiority in the semiring notation (with  $\otimes$ ) Moreover, the 'corresponds' is wrong, since superiority generalizes non-negative edge weights
- ☐ PS: Knuth's definition of superiority includes monotonicity
- ☒ 128: increaseS
- ☐ 136: what implies?
- ☐ 143: Lemma 4 is Lemma 3 in [21]
- ☒ 146: need infinite sums with  $\oplus$  -> need to extend  $\oplus$  to infinitely many operands
- ☒ 147: dom is undefined; I can also be  $\mathbb{N}$
- ☒ 151: partitionS
- ☒ 152: I'd use plural, i.e. products and sums
- ☐ 153: quantify  $x$  and the family  $(y_i)$  for  $i \in \mathbb{N}$
- ☐ 155--156: alphabets are usually finite and non-empty sets
- ☒ 156: the sentence ends after ... which is difficult to spot
- ☐ 159: denoted by; also it seems like  $\bar{\Phi}$  is a  $(\Sigma \cup \Sigma \times \Delta)$ -indexed family, why don't you introduce it like this
- ☐ 161--171: use an itemize environment
- ☐ 162--165: I think these should also hold for  $\Delta$ , why don't you write this explicitly?

- ☒ 186: calculability -> computability?
- ☐ 188: the definition of  $\oplus_{\Delta}$  can be seen to be implicit. Please fully quantify the alphabets of a label theory
- ☐ 195: references: there is earlier research on symbolic automata, e.g. Veanes (2013)
- ☒ 196: put respectively after output
- ☒ 199: use \colon rather than : after function names (this also applies to several other places in the paper)
- ☒ 203: THE number
- ☐ 208: What does  $T$  compute?
- ☒ 216: by convention -> by definition
- ☐ 219: resp.\@
- ☐ 220--221: you mean that a term CONTAINING  $w_{ij}(\dots)$  equal to 0 is ignored
- ☒ 228: three superfluous commas
- ☒ 228--229: with  $\oplus/\otimes$  -> in terms of  $\oplus/\otimes$
- ☒ 232: over THE alphabets
- ☒ 237: superfluous commas
- ☐ 238: text in the margin, also it is not clear whether the if refers to the entire line or just the right definition
- ☒ 247: I wouldn't define out just inside parentheses
- ☒ 251: based ON
- ☒ 252: for EVERY
- ☒ 253: replace comma by and
- ☒ 281: Partitioned
- ☐ 284:  $x$  is a variable, so it should be typeset in math mode
- ☒ 290: appoggiatura; ornamental; also, I'd give this explanation already in Example 1 where the appoggiatura is first used
- ☐ 291: I'd at least say once that  $\Delta_i$ ,  $\Delta_c$ , and  $\Delta_r$  are pairwise disjoint (or explain the symbol)
- ☐ 297: the extension here is not as in Section 3, there, the different function hat at least similar types
- ☐ 299: every line expands into the margin; moreover, in the rightmost equation of line three, in the application of  $w_c$ ,  $p'$  and  $q'$  are swapped
- ☒ 310: topupmost -> topmost
- ☐ 316: equation extends into margin
- ☒ 330--331: as indicated by the stack top: better explain that stack symbols are time intervals
- ☐ 331: I think the application of  $w_c$  is wrong (same with  $w_r$  and  $w_i$  later), shouldn't the : be a comma and what is  $q$  in the middle?
- ☐ 336: how are states (which are in  $Q$ ) pushed onto the stack if  $\Gamma = \Delta_c \times P$ ?
- ☐ 339: what are  $j$  and  $c$ , do you mean  $i$  and  $d$ ?

- ☐ 341: shouldn't there be a transition for returning from a measure?
- ☒ 345: exist (singular)
- ☐ 348: for? do you mean accepted by? but the language of an swA isn't defined either
- ☒ 352: Dijkstra-like algorithm?
- ☐ 354: readers may have an easier time seeing this argument if you say that  $S$  is extremal. You should say in the preliminaries that  $S$  is extremal if and only if  $\leq_{\oplus}$  is total.
- ☒ 366: be popPED
- ☐ 371: How is it ensured that the symbol  $\langle c, p \rangle$  isn't popped and pushed again later during the computation?
- ☒ 375: converge (singular)
- ☐ 376: I do not understand the and at termination, and its correctness part. What do you want to express?
- ☐ 379: Almost all update equations need to compute infinite sums. Don't you think that the assumption of  $\bar{\Phi}$  being effective is pretty strong and gives you the result "for free"? I think you should at least elaborate on this and say to what degree your effective label theory is a generalization of the decidable label theory of unweighted symbolic automata.
- ☒ 386: We assume the following input to be given
- ☒ 394: the notation "sw-parsing" is undefined
- ☒ 416: word order: put "to  $T$ " in the end
- ☒ 420: "the sw-parsing problem" or "sw-parsing", but not "the sw-parsing"
- ☒ 424: the triple use and does not work well
- ☒ 431: double the
- ☒ 435: there is something missing between "here" and "offline"
- ☐ 629: typical references for regular tree grammars are Brainerd (1969) and Gecseg and Steinby (1984)
- ☒ 647: inn  $\rightarrow$  in
- ☒ 650: missing period after function
- ☒ 652: I think  $\bar{w}$  should be  $\bar{w}'$
- ☐ 654: an intuition for the  $\bar{q}$  would have made understanding the construction far easier

Original literature:

- Herrmann and Vogler (2016): L. Herrmann & H. Vogler (2016): Weighted Symbolic Automata with Data Storage. In: Developments in Language Theory, Springer, pp. 203–215, doi:10.1007/978-3-662-53132-7 17.
  - Herrmann (2020): L. Herrmann (2016): Weighted Automata with Storage. PhD thesis. Technische Universität Dresden, 2020, urn:nbn:bsz:14-qucosa2-740685
-

## REVIEW 2

---

SUBMISSION: 89 TITLE: Symbolic Weighted Language Models and Quantitative Parsing over Infinite Alphabets AUTHORS: Florent Jacquemard, Philippe Rigaux and Lydia Rodriguez de la Nava

----- Overall evaluation ----- Transducers and visibly pushdown automata are used for parsing input words into abstract syntax trees. When the input words carry information from an infinite domain, such as time stamps, the transducers and automata need to be extended to handle these, which is the topic of this paper. The transducers are extended in one direction to handle inputs from the infinite domain, using the concepts from symbolic automata. Transducers are extended in another direction to give multiple outputs with associated weights. These can be used to produce multiple options for intermediate representations before computing the abstract syntax trees. The multiple intermediate representations can be compared and the one with the minimum weight can be picked up for further processing by visibly pushdown automata to construct the abstract syntax tree. The visibly pushdown automata are again extended to handle symbolic inputs and output weights for different possible runs, which represent different possible abstract syntax trees.

The contributions of this paper are extending transducers and automata to handle symbolic inputs and to output weights in a semiring instead of Boolean values. The paper also gives procedures for computing minimum weights for given inputs and to compute witnesses for these minimum weights, which will be helpful to construct syntax trees. The definitions have been extended from the respective symbolic and weighted models. The technical details are mostly well written and complemented with intuitive explanations using a running example. However, for now, I am still giving weak accept instead of accept, since I am having difficulty understanding a few points. See the last two comments below.

- ☐ Line 209: "T is based on an intermediate function ...": should this be "The semantics of  $T$  is based on an intermediate function ..."?
- ☒ Line 281: "artitioned into three" -> "partitioned into three"
- ☐ Last line of algorithm 1: What exactly is meant by "update  $d_{\perp}$  with  $\langle q_1, q_2 \rangle$ "? Does it mean update the image of  $\langle q_1, q_2 \rangle$  under  $d_{\perp}$  with the new value as computed in Figure 3? Write this clearly.

Figure 3: This figure is supposed to compute  $\langle q_1, q_2 \rangle$ .

- ☐ First of all, it is not clear what is meant by "compute  $\langle q_1, q_2 \rangle$ "; I am guessing, as I wrote in the comment above that the computed quantity is the image of  $\langle q_1, q_2 \rangle$  under  $d_{\perp}$ . However, even this doesn't make sense; there are some series of equations written, for  $d_{\top}(q_1, p, q_3)$  (not for  $d_{\top}(q_1, p, q_2)$ ).
- ☐ Even in this equation, what is  $q_2$ ? Do we take the minimum (wrt  $\oplus$ ) over all  $q_2$ ?
- ☐ The second line says  $d_{\perp}(q_1, p, q_3)$ , which doesn't make sense because  $d_{\perp}$  takes two arguments, not three.
- ☐ There are two equations for  $d_{\top}(q_1, p, q_3)$  and one for  $d_{\top}(q_0, p, q_3)$ . How are these related? It is very confusing what is being computed here and how. Please update this to make it clear.

## REVIEW 3

---



----- Overall evaluation ----- **Review - "Symbolic Weighted Language Models and Parsing over Infinite Alphabets"**

Overall the paper is good, though somewhat in need of editing, spelling correction etc.

It describes how to usefully incorporate weights into symbolic automata, and how to leverage existing algorithms over symbolic automata and transducers to achieve a form of parsing of strings over infinite alphabet. In particular, the running example can take into account both linear fit along a continuous interval, as well as the complexity of the inferred syntax tree.

The method is described as empirically useful, and theoretically novel.

The descriptions of the various definitions and proofs are mostly clear and correct, apart from typos, and combines ideas from several previous works into a set of formalisms that are both independently interesting, and combine to form an efficient weighted parsing algorithm for VPL over infinite alphabets.

Comments of a more editorial character follows:

**Introduction:**

- ☐ The literature review should also include work by Hermann and Vogler, who bring in weights into symbolic automata as well, as well as stacks and other types of data storage, though their focus is more on exploring theoretical properties than empirical implementation, and make no complexity claims, as far as I can see. (Herrmann, Luisa, and Heiko Vogler. "Weighted symbolic automata with data storage." International Conference on Developments in Language Theory. Springer, Berlin, Heidelberg, 2016.)

On lines 72+, the authors refer to Mohri in saying that "[...] given an input word  $s$ , the *SW-parsing* problem aims at finding  $t$  minimizing  $T(s, t) \times A(t)$ , called the distance between  $s$  and  $A$ [...]"

- ☐ It is not entirely clear to me what part of this expression they consider to be the distance as described by Mohri. Is it the distance, as defined by  $T$ , from the  $s$  to the distribution over ASTs  $t$  defined by  $A$ ? If so, it may be useful to structure the sentence somewhat differently, e.g. "[...] finding the linearized AST  $t$  minimizing  $d = T(s, t) \times A(t)$ , where  $d$  denotes the distance between  $s$  and the distribution over linearized ASTs described by  $A$ , using the distance measure defined by  $T$ ."

Another possible interpretation would be that the distance is between the two distributions over linearized ASTs defined by, on the one hand,  $T$  partially applied to  $s$ , and on the other,  $A$ , but this does not seem to fit with the importance of the specific AST  $t$ , at least not with my understanding of Mohri's definition of distances between distributions.

I think that, for me, the confusion stems partly from Mohri exclusively talking about distances between two strings (languages, distributions) over the *same* alphabet, while  $s$ , crucially, comes from a *different* alphabet than the one over which  $A$  defines a distribution.

**Preliminary Notions:**

- ☐ The introduction of Label theories (l. 154++) does not depend on its first paragraph containing the definitions of words and alphabets, as far as I can tell, so I would let this first paragraph sit under its own title of e.g. "Alphabets and words".
- ☐ It is also, separately, not clear to me what the intuition and motivation is for using a label theory, at least at this point in the paper. In particular, I am unsure why any of the specific closures in ll 161+ are considered necessary. It may be a better use of space to define label theories rather simply, here, and put the specifics of the closures into the appendix, together with the various properties already present there.

## SW Automata and Transducers

- ☐ A brief explanation of what you consider Mohri's approach to the computation of distances between words and languages may be helpful.

The paragraph starting at line 224 is a great summary of the way you consider edit-distances in this paper.

- ☐ In Example 8, you disallow the last note being an error - is there some specific justification for that, or is it just for simplicity?

## SW Visibly Pushdown Automata

Example 14 is quite limited in detail in comparison to the earlier examples, while being more complex in scope. This makes the specifics of the example hard to double check, and to properly understand.

- ☐ It is also somewhat difficult to separate the symbols that form part of the language from those of the meta-language in the specifications of the various  $w$  function calls.
- ☐ Additionally, you write "the above transition pushes the state  $q_{i+1/c}$  on[to] the stack" - I would encourage you to choose different letters and terminology than "state  $q$ " for stack symbols, to reduce the risk for the reader confusing the stack symbols and states.
- ☐ In fact, it may be a good idea to show a fuller schematic of a few steps of the sw-VPA computation, with relevant connections between various variables highlighted, what equalities need to hold etc.
- ☐ On line 366 of Proposition 16, the "can be" should be "must be", surely, since what you are describing is a computation that should leave the original stack (exactly) untouched?
- ☐ It seems to me that there is a consistent lack of quantification over  $P$  in the section starting with the definition of  $b_{\top}$ . This is not hugely problematic for the proof, since the algorithm is still defined (mostly) properly, and the complexity should only be impacted by a factor of something like  $|P|$ .

## Minor comments:

- ☒ l. 87. I would front "input" in the sentence, yielding "a given input *timeline* of [...] alphabet  $\Sigma$ , is parsed into[...]"
- ☒ l. 90. Please parenthesize the pitch-timecode pairs for readability. As an aside: these are strictly the onsets? And strictly monophonic?

- ☐ l. 100. Shouldn't the first measure end before the second starts? This would make  $O$  a forest rather than a tree - easily fixed with a "piece" root element.
- ☒ l. 156. The ellipsis between  $\Delta$  and  $\Sigma^*$  confuses the sentence boundaries for the reader. Better to go "The set of finite sequences (*words*) over  $\Sigma$  is denoted  $\Sigma^*$ , while  $\varepsilon$  denotes the empty word[...]".
- ☐ l. 220. The sentence starting here is somewhat confusing (and possibly wrong, depending on if  $q'$  and  $q''$  are intended to be distinct). I suppose that the point is that one can quit the recursion "early" whenever  $w_{ij}$  returns 0, but the language could be clearer, something like "Since 0 is absorbing for  $\times$ , and the identity element for  $+$  in  $S$ , if  $w_{ij}(q, a, b, q'')$  is 0 (meaning there is no transition from  $q$  to  $q'$  reading  $a$  and  $b$ ) that entire term can be ignored in the sum."
- ☒ l. 240: an fixed weight -> a fixed weight
- ☒ l. 251: based with an intermediate -> based on an intermediate
- ☒ l. 276: generalizing themselves -> themselves generalizing
- ☒ l. 281: artitioned -> partitioned
- ☐ l. 288: same as on line 100, measure 1 should maybe end before measure 2
- ☒ l. 290: ornemental -> ornamental
- ☒ l. 306: reads and pop from stack -> reads and pops from the stack
- ☒ l. 307: I would use "computations" or something like that rather than "transitions" here
- ☒ l. 310: topupmost -> topmost, though I'm not sure if it's necessary to describe the basic workings of a stack. If you choose to do so, then also
- ☒ l. 311: content -> pair
- ☒ l. 323: particular -> special
- ☒ l. 365: "case of in the" -> either "case in the" or "case of the"
- ☒ l. 366: pop -> popped
- ☒ The second case in Figure 3 should start with  $d_{\top}$ , not  $d_{\perp}$
- ☒ l. 395: I would omit "output". If you want to draw attention to the different alphabets involved, I would say "between the string  $s$  over the input alphabet, and the weighted language over the output alphabet defined by  $A$ "
- ☒ l. 399: correspondance -> correspondence
- ☒ l. 402: "weight a second time division" -> something like "weight assigned to an additional second time division"
- ☒ l. 416: extra commen between " $T$ " and "of new transition"
- ☒ l. 417: Corolary -> Corollary
- ☒ l. 420: This whole paragraph should be shaped up, writing wise.
- ☐ l. 434:  $swM??$
- ☒ l. 435: "offline" -> "works offline"
- ☒ l. 437: extraneous comma after "such as"

#### References:

- ☐ Ref [22] and [23] both seem to refer to the same work by Mohri

