Web Speech API

Julius Adorf
KTH Royal Institute of Technology, Stockholm

May 27, 2013

Abstract

This technical report describes the current state of the Web Speech API, which provides a JavaScript interface for speech analysis and speech synthesis to web applications. The focus lies on measuring the performance of Google's speech recognition web service that is behind the experimental API implementation in the Chromium browser. To this end, sentence correctness and word accuracy is measured using the TSP speech database with more than 1400 recordings of the Harvard sentences. The report also puts the Web Speech API into historical context and speculates about future prospects. A practical use-case is given that demonstrates how the API can be used in a web application. Supplemental data and the use-case is published online.

1 Introduction

The Web Speech API is designed for speech analysis and speech synthesis. It allows web users to send speech input to web applications. The web applications use the Web Speech API to transform the speech into text. While mobile phone users have become used to a speech as a new input method, it is currently uncommon to see voice-controlled web applications.

There are good reasons why speech input in web applications might be beneficial for users. First, opening the field for a new input method enhances accessibility. Just like BrailleTouch enables visually impaired users to type without looking [1], speech input might provide a convenient means of alternative input. Second, speech is a hands-free input method. Just imagine a web application that uses hand gestures as input. Hand gestures conflict with standard mouse and keyboard control, whereas speech and hand gestures can be easily combined. Third, users have become more and more used to speech input on mobile phone applications and might demand that competing web applications have the same capabilities. In these times, the distinction between desktop applications, web applications, and mobile phone application blurs. This development is mirrored on the market of electronic devices, ranging from notebooks with touchscreen via tablet computers of different sizes to a wide range of mobile phones. In such an environment, web applications that accept a wide range of input methods might gain an edge over competitors. In summary, speech input in the web might enhance the experience of individual users, might allow web applications to use multiple input methods, and might add what the user already expects.

The Web Speech API is an experimental JavaScript API. The experimental nature is a reason why the introduction so far has remained mostly speculative. A core feature of the API is that the speech recognition itself is largely transparent to the web developer. the actual speech recognition is delegated to a web service. The web developer does not directly interact with the web service but rather communicates with the user agent through events. It is the responsibility of the user agent to implement the interface and the communication between the user agent and the speech recognizion web service.

The Web Speech API is developed by the W3C Speech API Community Group. The initiative is open to everyone but it appears that it is largely driven by two companies: Google and Openstream. It is a recent initiative and the specification draft awaiting final commitments was published only in October 2012. There is a vendor-specific implementation of the API draft available in Chromium with version 25 or newer, despite of the specification neither being finalized nor belonging to a W3C Standards track.

This report evaluates both the Web Speech API and the speech recognition service used by Chromium by default. Section 2 delves into the history of the API and what influenced the design decisions. Section 3 describes the API itself. Section 4 looks at the current browser support. Section 5 attempts to measure the performance of the speech recognition web service used by Chromium. Section 6 adopts the technology and presents a web application with speech input. Finally, Section 8 goes full circle and speculates about future developments in how speech recognition might change the world of web applications.

2 History

Work on the Web Speech API can be traced back at least to the end of 2011. developed The Speech API Community Group currently comprises 32 members. Most of the members who specified their affiliation on their profile pages on the community wesite are Google employees (see Table 1). Given that there are at least five Google members in the group, it seems likely that Google has a particular interest in such an API.

Google	5
Openstream	3
Others	14
N/A	10
Total	32

Table 1: Affiliations within the Speech API Community Group.

3 API Specification

This section discusses the Web Speech API itself. First, its scope is explained. Then, its design is presented. Finally, the supported features are listed. The latter part is of particular interest for both API users and the providers of web services for speech recognition because the functionality that is exposed by the interface also needs to be implemented by the web service.

The Web Speech API covers both speech analysis and speech synthesis. In other words, it supports the conversion of speech to text and vice versa. The API is purely in JavaScript, which is currently one of the predominant client-side scripting languages of the web.

The Web Speech API is event-based, which fits in well to the rather callback-heavy style of coding with JavaScript. Calls to the API are handled by the user agent which in turn takes charge of all communication with a web-based speech recognition service. This of course requires that the user agent implements the API. The event-based architecture allows programs to asynchronously process speech. Events are also used to report intermediate speech recognition results which is convenient because it allows programs to give almost immediate feedback to the user. Speech recognition can be interrupted at any time, which is convenient because it relieves the web developer of extra work in the event handler routines.

In order to be able to serve users all over the world, the API must support different languages. It is possible to set the language for speech recognition. By default, the language is defined by the locale settings of the user agents. The language has to be specified a-priori to speech recognition. Hence, it must be known in advance which language is expected. It is not possible to freely mix languages.

The API defines ways to adapt the speech recognition

system for specific tasks. A grammar can be given to the speech recognition system. The system can possibly take advantage of the constraints given by the grammar for improved speech recognition. It is written in the specification that the grammar format is still subject to discussion and not yet finished.

Intermediate or final recognition results are given in the form of several candidate sentences, each associated with a certain confidence value. The most likely transcription is listed first. The API distinguishes between parts of a transcription that are preliminary and parts that are final. This is useful when looking at intermediate results while speech recognition is ongoing.

Finally, privacy issues need to be mentioned. As the Web Speech API shall not be abused in order to spy on the user by secretly listening to microphone input, the user needs to be asked for permission first. This problem becomes apparent in the use case presented in Section 6.

4 Browser support

Currently, there are five popular browsers Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, Safari, and Opera in use [2]. However, only Google Chrome (version 25+) has experimental support of the Web Speech API. The Web Speech API is not a W3C Standard. Given the experimental nature, the symbols in the implementation are vendor-prefixed at the time of writing, and the code using the Speech API in Chromium currently looks like this:

```
var rec = new webkitSpeechRecognition();
recognition.onresult = function(event) {
  // ...
};
recognition.start();
// ...
```

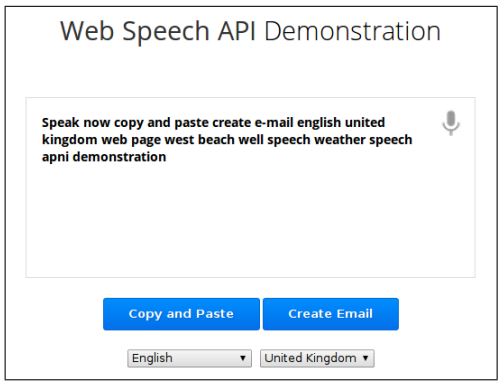

Figure 1: A web page by Google that demonstrates the Web Speech API. The user can dictate words, and the recognized words are displayed in the text area while the user is dictating.

5 Evaluation

The usefulness of the Web Speech API is limited by the performance of the underlying web service that performs speech recognition. The performance of this web service is evaluated in this section. First, the solution for the the technical challenges are explained because it is not straight-forward to use the Web Speech API for other purposes than user input. Then, the experimental setup and the dataset are described. Finally, the results are discussed.

5.1 Technical preparations

The Web Speech API allows users to record audio from the microphone, which is then sent via a HTTPS POST request to the speech recognition web service. The result can then processed within the limits of a JavaScript application running in the Chromium browser. For evaluational purposes it is advantageous to access the web service directly such that recorded audio data from files can be passed to the web service and the results processed without the restrictions of the browser's sandbox. The solution is therefore to open a HTTPS connection, upload the audio data via a POST request and retrieve the recognition re-

sults for further processing. The source code of the Chromium project provides all necessary information for constructing such HTTPS requests. There are articles on the web that describe the necessary steps [3]. It is not clear how Google is positioned towards this particular hack of its speech recognition web service.

5.2 Benchmark dataset

The *TSP Speech Database* [4] serves as a benchmark in this evaluation. It is one of the few annotated speech recognition datasets that is available in the world wide web at no cost [1]. The database contains more than 1400 utterances from 12 male and 12 female speakers. In total, there are 720 sentences distributed over 72 lists of 10 sentences each. This list is known as the *Harvard sentences* [5]. The recordings were performed in an acoustic anechoic room and designed to have a low noise level. All sentences are in English, and most of the speakers were adult native speakers. The reader is highly recommended to listen to some of the recordings in order to fully appreciate the quantitative evaluation that follows.

5.3 Experimental setup

This section describes the experimental setup and equips the reader with what is necessary to interpret the results presented in Section 5.4. Concretely, this section explains how the dataset is sent to the web service and how the recognition results are compared to ground truth.

The basic idea is rather simple: each sentence in the TSP Speech Database is separately sent to the speech recognition web service. It appears that the web service accepts audio data in the FLAC [6] format. Conversion from the wave-encoding of the dataset to the FLAC encoding required by the web service is lossless, which is fortunate for the purpose of evaluation. In order not to bombard the Google web service (don't be evil to put it the Google way), the sentences are sent to the web service over a period of 24 hours, which corresponds to one sentence per minute. The returned transcriptions are finally compared to ground truth. Results are reported on a sentence level and on a word level.

[1] This is a student report without funding.

edit	cost
insertion	7
deletion	7
substitution	10

Table 2: The edit operation costs used by default in the sequence alignment function of the Hidden Markov Model Toolkit.

Reporting results on a sentence level is straightforward. Here, it is enough to see whether a transcribed sentence and the corresponding test sentence match. However, reporting becomes more complicated on a word level because the speech recognizer might transcribe words that were not spoken (*insertion*), might not transcribe words that were spoken (*deletion*), or confuse words (*substitution*).

This evaluation uses sequence alignment in order to compare the results on the word level. The Levenshtein edit distance between the transcribed sentence and the test sentences is computed. This edit distance is the minimum cost for word insertions, word deletions, and word substitutions that align the transcribed sentence and the test sentence. From the optimal alignment, we can compute the word accuracy W_{acc}. Let N denote the number of words in the test sentence. Let D be the number of deletions, S the number of substitutions, and I the number of insertions in the optimal alignment. Then, the word accuracy W_{acc} is computed by:

$$W_{\text{acc}} = \frac{N - D - S - I}{N} \tag{1}$$

Differences in case, whitespace, and punctuation are not taken into account. The punctuation problem is for example treated in [7], but not part of this evaluation. The value of the word accuracy measure for evaluational purposes is subject of discussion in the literature [8].

Table 2 lists the costs used in this evaluation – the same costs as the Hidden Markov Model Toolkit (HTK) [9] uses by default. The word accuracies reported in this evaluation can be directly compared to the word accuracies reported by the HTK command-line tool `HResults`.

5.4 Results

The results of the experiments are first discussed on a sentence level, and then on a word level. Details are given in two tables in the appendix. Table 4 shows the results broken up by speakers, whereas Table 5 groups the results by list.

Only a minority of all spoken sentences is recognized correctly, where correctly means that the test sentence and the transcribed sentence match in every single word. From a total of 1444 spoken sentences, 306 sentences are correctly recognized. In other words, 21% of the spoken sentences are recognized without insertion, deletion, or substitution errors. Whether the speaker is male (21% correct) or female (20% correct) makes no real difference. 32% of the 66 sentences spoken by the two children are recognized, but a conclusion can hardly be drawn given the small sample size.

Altogether, the speech recognizer has difficulties with many of the sentences. The sentences are difficult, at least when judged by a human listener. They sound uncommon and are sometimes difficult to understand for non-native English speakers. How this relates to the difficulty level for machine understanding is out of scope of this work. However, here is an example from list 69 where the speech recognizer performs particularly poorly (five substitutions and one insertion): the sentence `"the steady drip is worse than a drenching rain"` was mistaken for `"the city trip is worth the trenching rain"`. Table 3 presents ten randomly selected spoken sentences and the corresponding speech recognition results. Both good and bad results can be seen.

On the word level, results look brighter than on a sentence level. The speech recognizer correctly recognizes 8540 words out of a total of 11540 spoken words. This means that 74% of all spoken words are correctly recognized. In contrast to the percentage of correct words, the word accuracy takes insertions into account. However, only the number of insertions is low and the overall word accuracy of 73% does not differ much from the percentage of correctly recognized words. The word accuracy does not really differ between males (74%) and females (72%). Again, the only two children scored higher (83%).

The detailed statistics are reported in Table 4 and Table 5. Both table present the same data, but grouped differently.

The speakers in Table 4 are identified by a two-letter combination, where the first letter indicates whether the speaker is a male (M), a female (F), or a child (C). The genders of the two children are not reported. The second column in lists the number of sentences spoken by the respective speaker. Most of the speakers spoke exactly 60 sentences, but due to some mistake in the dataset acquisition [4], some recordings got lost.

The recordings of the sentences in the TSP Speech Dataset are organized in the 72 lists of the Harvard sentences. Table 5 groups the results by list. While most of the sentences in each list were spoken exactly twice (once by a male and once by a female), the lists 1–6, 14, and 25–30 are irregular.

The last six columns of Table 4 and Table 5 have the same interpretation. The third column lists the percentages of sentences that are correctly recognized. The fourth column lists the number of words (N) in the test sentences. Columns five, six, and seven report the number of insertions (I), deletions (D), and substitutions (S), respectively. The last column provides the word accuracies obtained from the previous four columns using Equation 1.

6 Use Case

It was stated in the introduction that speech input might be beneficial to web applications. This section presents the particular use case of a simple to-do list

Test sentence	Transcribed sentence	W_{acc}
He wrote his name boldly at the top of the sheet.	he wrote his name boldly at the top of the sheets	90%
Tend the sheep while the dog wanders.	1000 the dog wanders	42%
The slush lay deep along the street.	this Leslie Depot Long Street	14%
Their eyelids droop for want of sleep.	O'Reilly's group for want to sleep	28%
Time brings us many changes.	time brings us many changes	100%
The cup cracked and spilled its contents.	the cup cracked and spilled its contents	100%
Pages bound in cloth make a book.	pages: found in cloth make a book	85%
Green moss grows on the northern side.	remind girls in the northern side	42%
Hoist the load to your left shoulder.	push to look to your left shoulder	57%
The source of the huge river is the clear spring.	the source of the huge River is the clear spring	100%

Table 3: Ten randomly selected speech recognition results.

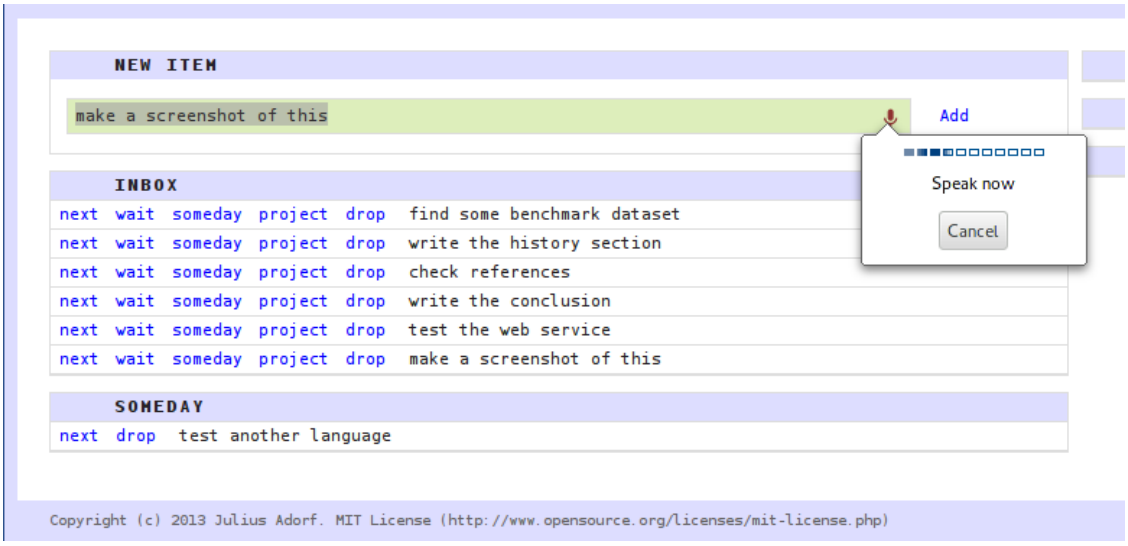

Figure 2: A simple to-do list application that features speech as an alternative input method.

application, and along with it some practical experiences that were gathered during implementation and experimentation.

The simple to-do list application (see Figure 2) allows the user to add to-do items to a list. These to-do items can be moved between several lists by drag and drop. Speech input is particularly beneficial for such an application. Perfect accuracy is not required because to-do lists are personal. Humans can easily recover the meaning of their own to-do items even if they contain a misspelling or a wrong word due to a mistake made by the speech recognizer. This can often be observed with users of pen and paper to-do lists. The written items are often almost illegible and incomprehensible without further context. As soon as the context disappears, the to-do item has usually also lost its importance. However, speed is all that matters because to-do lists are intended to make the user more productive, not the other way round. Given that error correction plays less a role in such a context, the hope is that speech recognition software becomes accurate enough that speech input is faster than keyboard input. The reader is invited to test the to-do list application on `www.dropandforget.com`.

Assuming that the users are surfing with Google Chrome 25+ (or Chromium), adding the speech input feature to the to-do list application is simple for developers. The only necessary change to the `HTML` page is to add the attribute `x-webkit-speech` to the input text field. Using Chrome 25+, the downside of this approach is that no interim results are shown in the input text field so that the user has to wait for the final result until he or she finished speaking. The upside is that the user is given feedback about the volume level while speaking and that no separate permission dialog needs to be taken care of.

An alternative approach to implementing speech input for Chromium is to access the Web Speech API directly. This enables instant feedback for the user, who can see the current maximum likelihood solution while speaking. However, this approach has a major downside with the current browser implementation because the user has to grant the web application permission every time the user wants to initiate speech input. Future browser implementations might feature a more sophisticated scheme for granting web application permissions to access the microphone.

7 Future

Research in speech recognition has been ongoing for a long time. Recently, it seems that the concept of Deep Neural Networks (DNN) has caused new dynamics in this field. Judging from the publications by Google Research, by Microsoft Research, by IBM Research and by the University of Toronto, Deep Neural Networks are reasons to hope for a leap forward.

These four research institutions recently published a paper that describes how Deep Neural Networks (DNN) were successfully used by the respective research groups to improve acoustic modeling [10]. The authors describe a novel generative pre-training stage as the key feature for the success, and as a reason why multi-layer networks gained interest after a long time of Hidden Markov Models with Gaussian Mixture Models dominating speech recognition research. In the pre-training stage, layers of the DNN are tuned without using the target values in the training set. It is good to see two approaches competing in speech recognition research.

8 Conclusion

In summary, this report reviewed several aspects of the Web Speech API. The history and the affiliated companies were presented. The API itself and the limited browser support were discussed. Google's speech recognition web service was evaluated. A small demo application showed how the API can be used. Current research at influential companies and university departments was reviewed in order to get an idea where research is heading.

As the reader may verify by him- or herself, as the evaluation showed, and as it is well-known, there is much room for improvement for speech recognition

systems. However, they are already sufficiently robust for many applications and are readily adopted by the users. Giving web developers an easy-to-use speech API helps spreading the technology.

This report contributed to both the field of speech recognition and web development. Recent state-of-the-art technology has been evaluated on the publicly available TSP Speech Dataset. Efforts have been taken to make the experiments repeatable and the results comparable. The dataset used for evaluation was difficult enough to pose a challenge for Google's speech recognition web service. A use-case was presented that shows how the Web Speech API can be applied in practice.

Mostly, this report focused on the speech analysis part of the API. The speech synthesis has not given much attention and is part of future work. The Web Speech API and the speech recognition technology is expected to advance continuously. It will be interesting to see how the speech recognition system performs in future versions. Optionally, improvements can be monitored by running the evaluation again at a later point of time.

References

[1] B. Frey, C. Southern, and M. Romero, "BrailleTouch: Mobile Texting for the Visually Impaired," in *Proceedings of the 2011 International Conference on Human-Computer Interaction*, 2011.

[2] w3schools.com, "Browser Statistics." `http://www.w3schools.com/browsers/browsers_stats.asp`. accessed on 2013-05-22.

[3] Mike Pultz. `http://mikepultz.com/2011/03/accessing-google-speech-api-chrome-11`, 2011. accessed on 2013-03-18.

[4] P. Kabal, "TSP Speech Database," tech. rep., McGill University, 2002.

[5] H. R. Silbiger and J. L. Sullivan, "IEEE Recommended Practice for Speech Quality Measurements," *IEEE Transactions on Audio and Electroacoustics*, vol. 17, no. 3, pp. 225–246, 1969.

[6] "Free Lossless Audio Codec." `http://flac.sourceforge.net`. accessed on 2013-03-18.

[7] A. Gravano, M. Jansche, and M. Bacchiani, "Restoring Punctuation and Capitalization in Transcribed Speech," in *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4741–4744, 2009.

[8] Y.-Y. Wang, A. Acero, and C. Chelba, "Is Word Error Rate a Good Indicator for Spoken Language Understanding Accuracy," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 577–582, 2003.

[9] "Hidden Markov Model Toolkit." `http://htk.eng.cam.ac.uk`. accessed on 2013-03-18.

[10] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

A Appendix

The appendix lists the detailed results obtained in the evaluation in Section 5. Supplemental material is published on the web.

Speaker	Sentences	Correct	Words	Ins.	Del.	Sub.	W_{acc}
CA	60	33%	466	4	5	65	83%
CB	6	17%	49	1	2	9	75%
FA	60	23%	466	3	21	79	77%
FB	60	8%	467	4	35	120	65%
FC	60	20%	475	8	32	83	74%
FD	60	18%	478	2	35	88	74%
FE	58	24%	461	4	23	84	76%
FF	60	23%	484	4	25	84	77%
FG	60	17%	456	2	52	122	62%
FH	60	15%	481	6	48	131	61%
FI	60	20%	495	3	31	102	72%
FJ	60	35%	497	0	15	65	84%
FK	60	15%	503	3	43	126	65%
FL	60	27%	480	6	25	102	73%
MA	60	22%	466	3	35	89	72%
MB	60	28%	467	1	19	84	78%
MC	60	15%	475	3	35	96	71%
MD	60	37%	478	4	12	62	84%
MF	60	13%	484	7	33	114	68%
MG	60	22%	456	2	27	82	75%
MH	60	25%	481	5	12	89	77%
MI	60	23%	495	8	25	92	74%
MJ	60	10%	497	3	46	144	62%
MK	60	12%	503	1	35	101	72%
ML	60	23%	480	4	32	84	76%

Table 4: Results of the evaluation grouped by the 24 speakers. See Section 5 for a detailed description of how to interpret this table.

Table 5: Evaluation results grouped by the 72 lists. See Section 5 for a detailed description of how to interpret this table.

List	Sentences	Correct	Words	Ins.	Del.	Sub.	W_{acc}
1	31	10%	251	3	4	59	73%
2	32	53%	257	0	7	25	86%
3	30	27%	243	0	15	37	79%
4	30	27%	219	2	15	40	73%
5	30	23%	228	0	9	37	79%
6	30	13%	225	6	11	39	74%
7	20	30%	172	1	8	29	77%
8	20	5%	162	0	10	30	75%
9	20	20%	140	1	8	32	71%
10	20	10%	154	0	16	43	62%
11	20	20%	148	0	4	37	71%
12	20	25%	158	3	8	33	72%
13	20	15%	158	2	16	27	71%
14	21	10%	167	1	9	37	71%
15	20	15%	154	4	11	35	67%
16	20	15%	158	2	15	33	68%
17	20	20%	166	2	10	26	76%
18	20	30%	154	0	6	24	80%
19	20	20%	152	2	2	29	78%
20	20	30%	164	0	6	22	83%
21	20	40%	166	0	13	22	80%
22	20	45%	158	0	7	15	86%
23	20	20%	158	2	8	36	71%
24	20	10%	158	2	11	26	75%
25	10	40%	81	0	2	7	87%
26	10	30%	73	0	2	9	86%
27	10	10%	78	0	6	17	70%
28	10	10%	81	0	1	18	76%
29	11	27%	89	3	12	23	59%
30	8	25%	68	1	2	12	78%
31	20	25%	166	0	7	33	77%
32	20	25%	156	3	11	35	69%
33	20	5%	158	0	18	40	62%
34	20	20%	162	4	6	31	74%
35	20	20%	168	2	8	31	76%

Table 5 – continues on the next page

Table 5 – continued from the previous page

List	Sentences	Correct	Words	Ins.	Del.	Sub.	W_{acc}
36	20	15%	158	2	8	28	75%
37	20	5%	156	0	17	40	63%
38	20	30%	148	0	11	27	74%
39	20	50%	144	1	6	15	85%
40	20	5%	152	3	14	37	64%
41	20	15%	156	0	17	37	65%
42	20	10%	156	0	14	48	60%
43	20	10%	156	5	7	34	69%
44	20	20%	156	1	13	39	64%
45	20	20%	158	0	11	35	71%
46	20	15%	168	4	10	51	60%
47	20	30%	164	1	8	23	80%
48	20	25%	160	0	11	38	68%
49	20	25%	178	0	4	22	84%
50	20	35%	168	2	13	31	73%
51	20	45%	168	1	5	19	85%
52	20	5%	164	3	16	52	55%
53	20	15%	144	1	8	30	73%
54	20	5%	168	4	10	40	67%
55	20	30%	176	0	9	30	79%
56	20	25%	162	1	7	36	72%
57	20	5%	162	1	16	47	62%
58	20	25%	186	1	10	39	73%
59	20	20%	156	0	9	23	80%
60	20	30%	152	0	10	34	72%
61	20	20%	164	0	11	33	72%
62	20	10%	166	2	17	31	70%
63	20	10%	178	0	7	44	71%
64	20	15%	164	1	9	38	68%
65	20	10%	160	0	24	41	59%
66	21	19%	182	1	10	40	72%
67	20	20%	162	1	9	30	75%
68	20	30%	156	1	11	29	74%
69	20	0%	158	6	14	46	57%
70	20	10%	186	1	14	38	73%
71	20	50%	150	0	3	17	87%
72	20	40%	148	1	6	26	80%