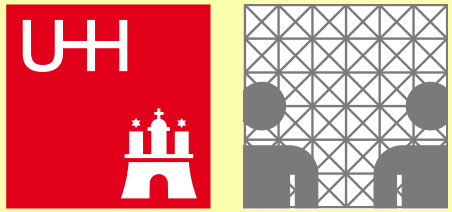


ESSLLI

Incremental Speech and Language Processing for Interactive Systems

Timo Baumann, Arne Köhn,
Universität Hamburg, Informatics Department
Natural Language Systems Division
{baumann,koehn}@informatik.uni-hamburg.de

Contents of the Course

- Monday:
  - introduction, major features of incremental processing
- Tuesday:
  - incremental processing for sequence problems
- Wednesday:
  - incremental processing for structured problems
- today:
  - generating output based on structured and partial input
- Friday:
  - wrap-up and outlook, also based on your questions and interests

Short Recap

- „true“ incrementality vs. restart-incrementality
- non-monotonicity allows to produce final output that is as good as a non-incremental processor's
- so far:
  - input side of a speech/language system
  - one type of input, one type of output
- today:
  - generate user-facing output from multiple types of input
  - limit span of non-monotonic operations

Short Recap II

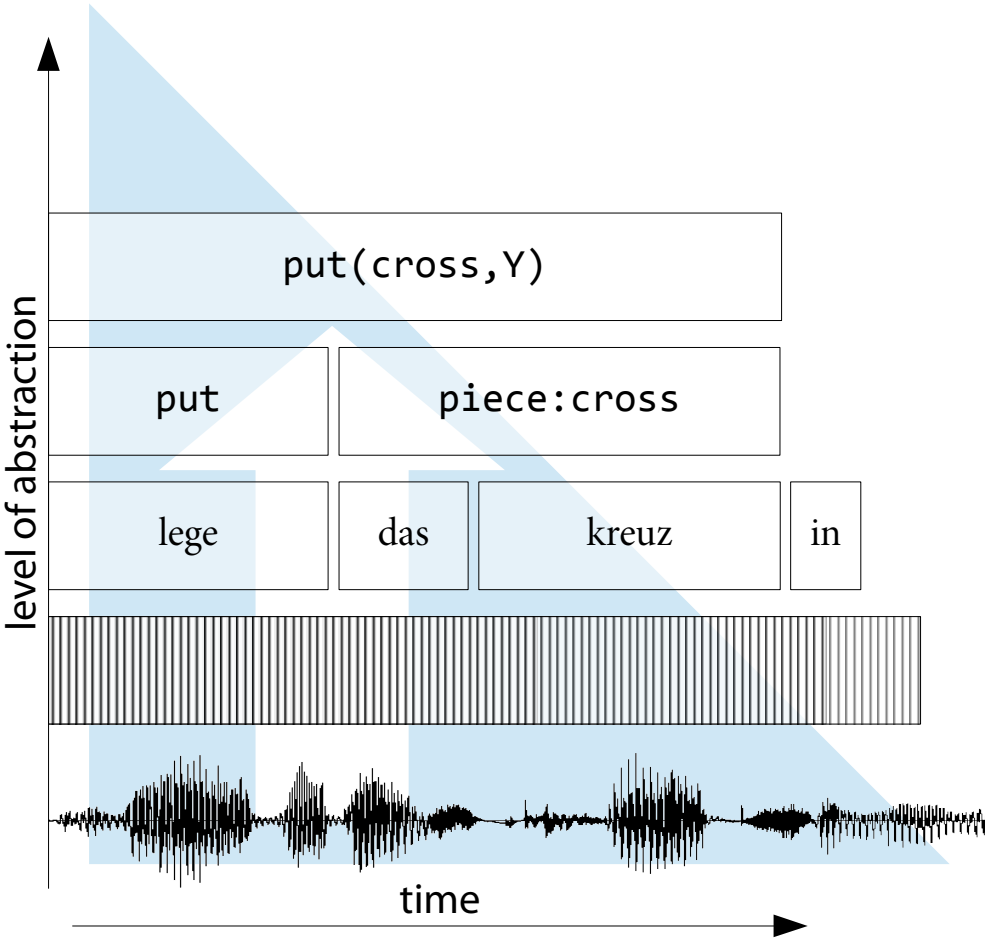
- **Lookahead**/context/latency: how much we allow the output to lag behind given constant extention of the input (generally: higher means more monotonic)
  - today: analyze when we require more/get by with less lookahead
- **Granularity**: size of the *minimal unit of processing* (generally: smaller is better)
  - today: when we have different types of input, we may have mixed granularity (as small as possible per-type)
  - mixed granulary can help to reduce lookahead requirements

- both lower lookahead and finer granularity help to reduce processing delays

Contents for today

- a (very short and sketchy) introduction to speech synthesis
- dealing with realtime pressure (which restricts non-monotonicity)
  - how much lookahead for acceptable non-incremental (i.e., post-hoc) quality?
  - how to organize the architecture to achieve concurrent processing
- mixed input types to improve overall performance

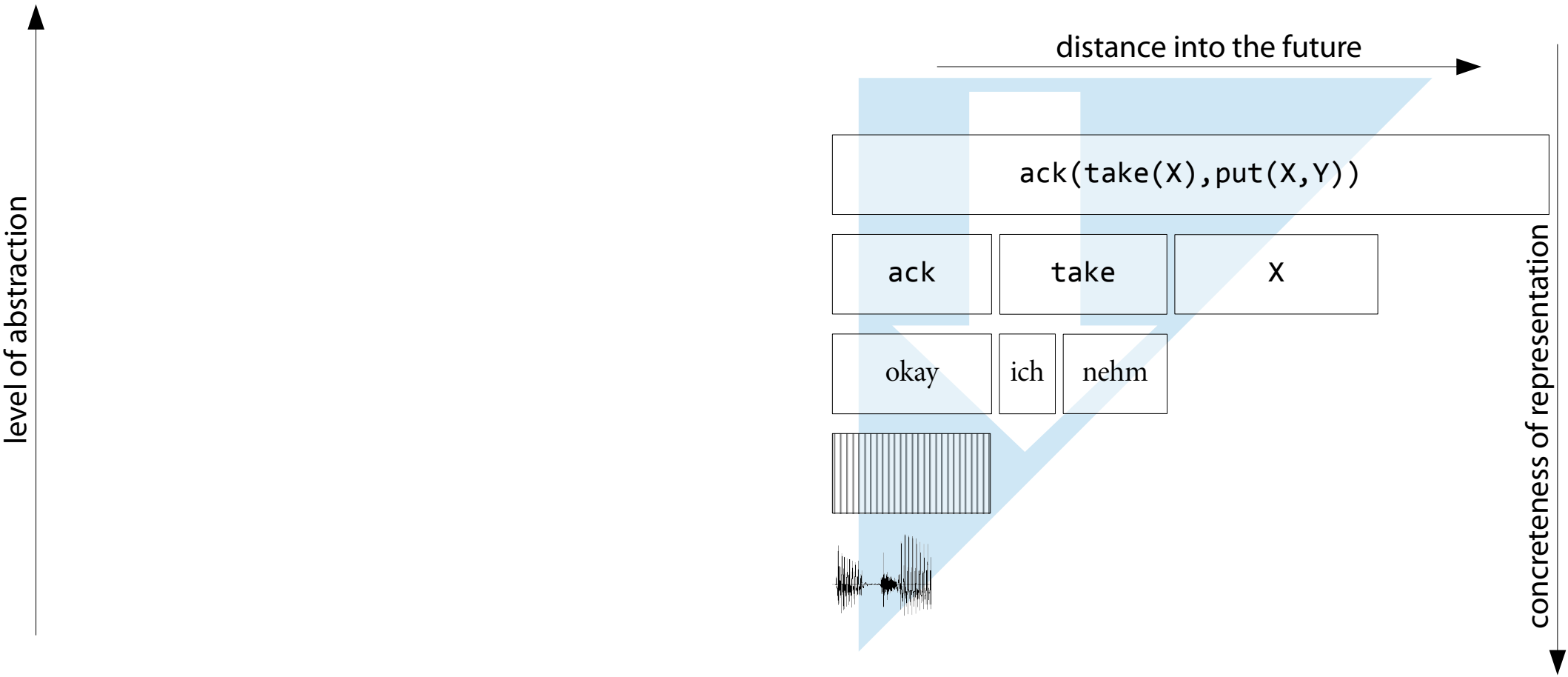
Consuming input incrementally



input side

Baumann (2013)

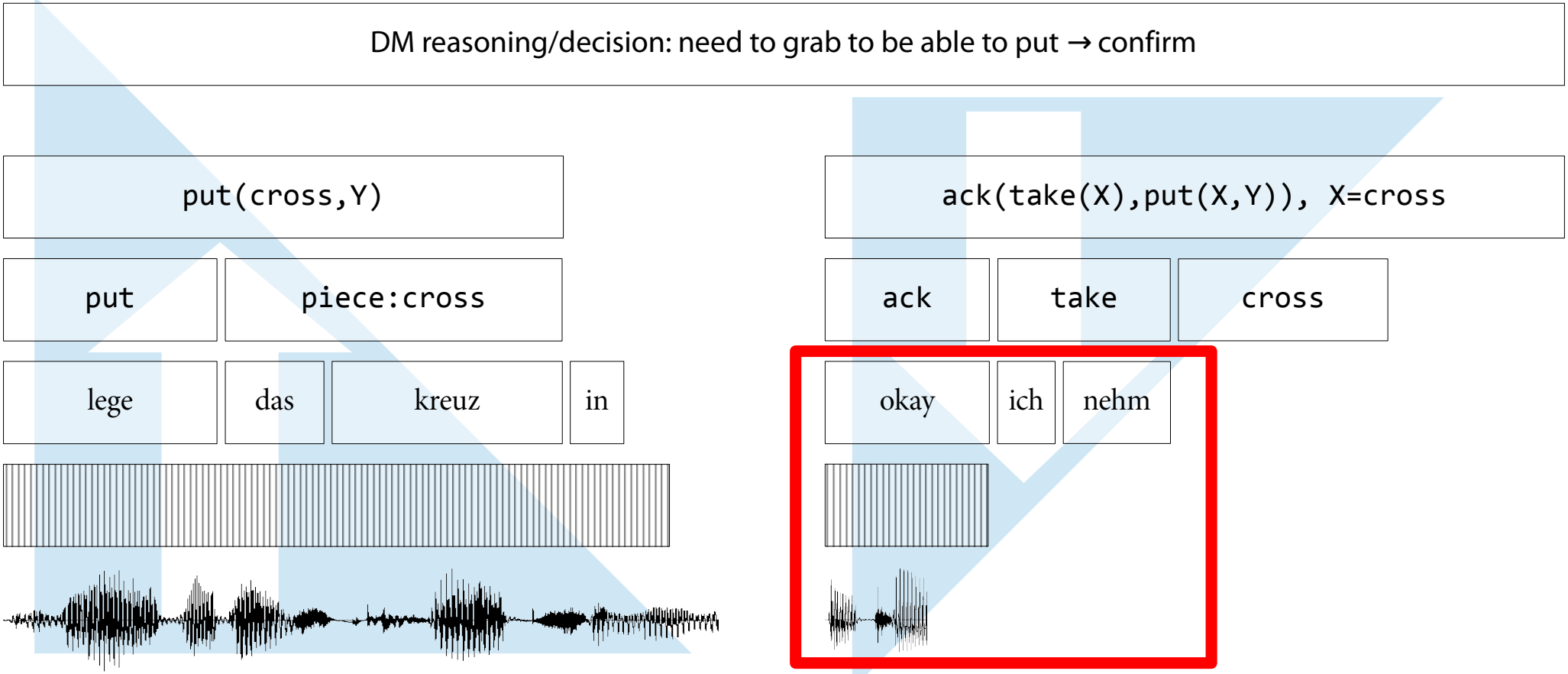
Producing output just-in-time



output side

Baumann (2013)

Decision making governs input/output combination

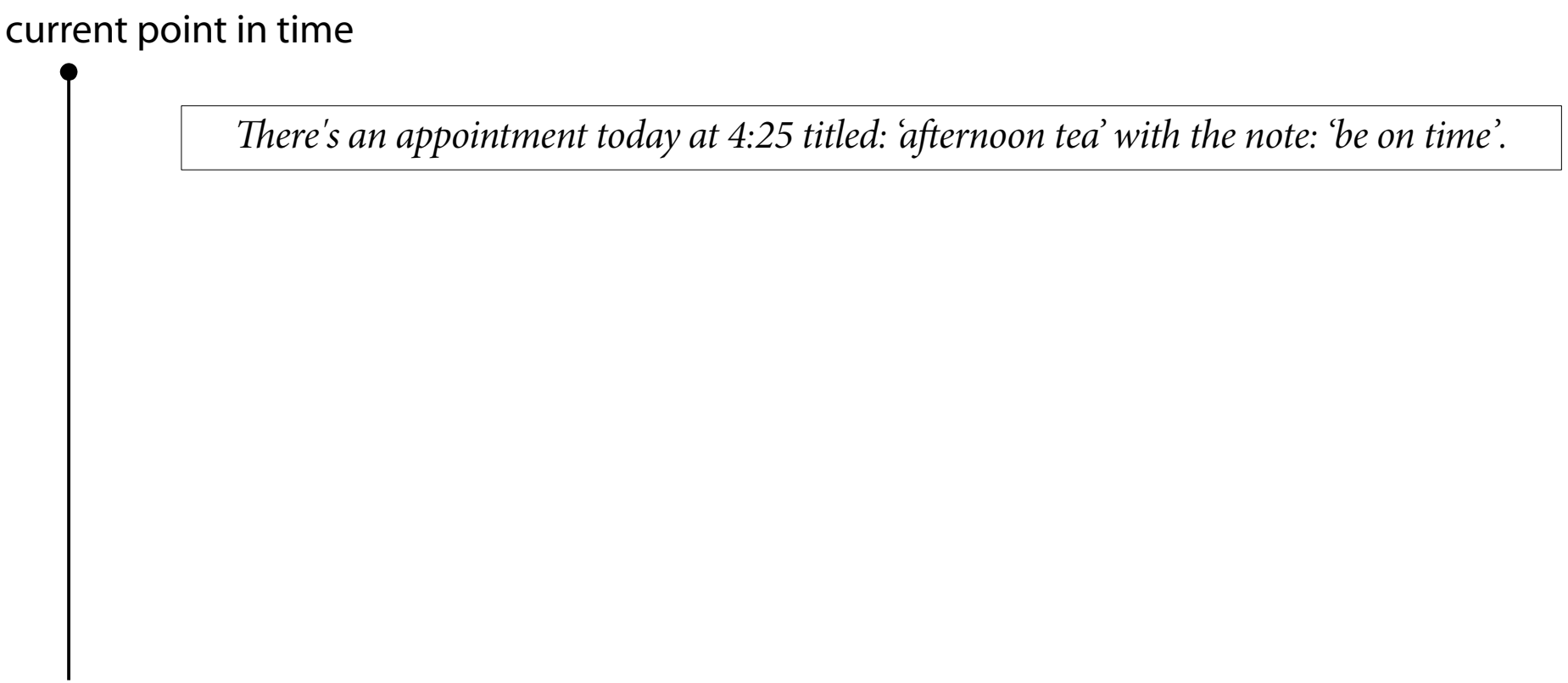


input side

output side

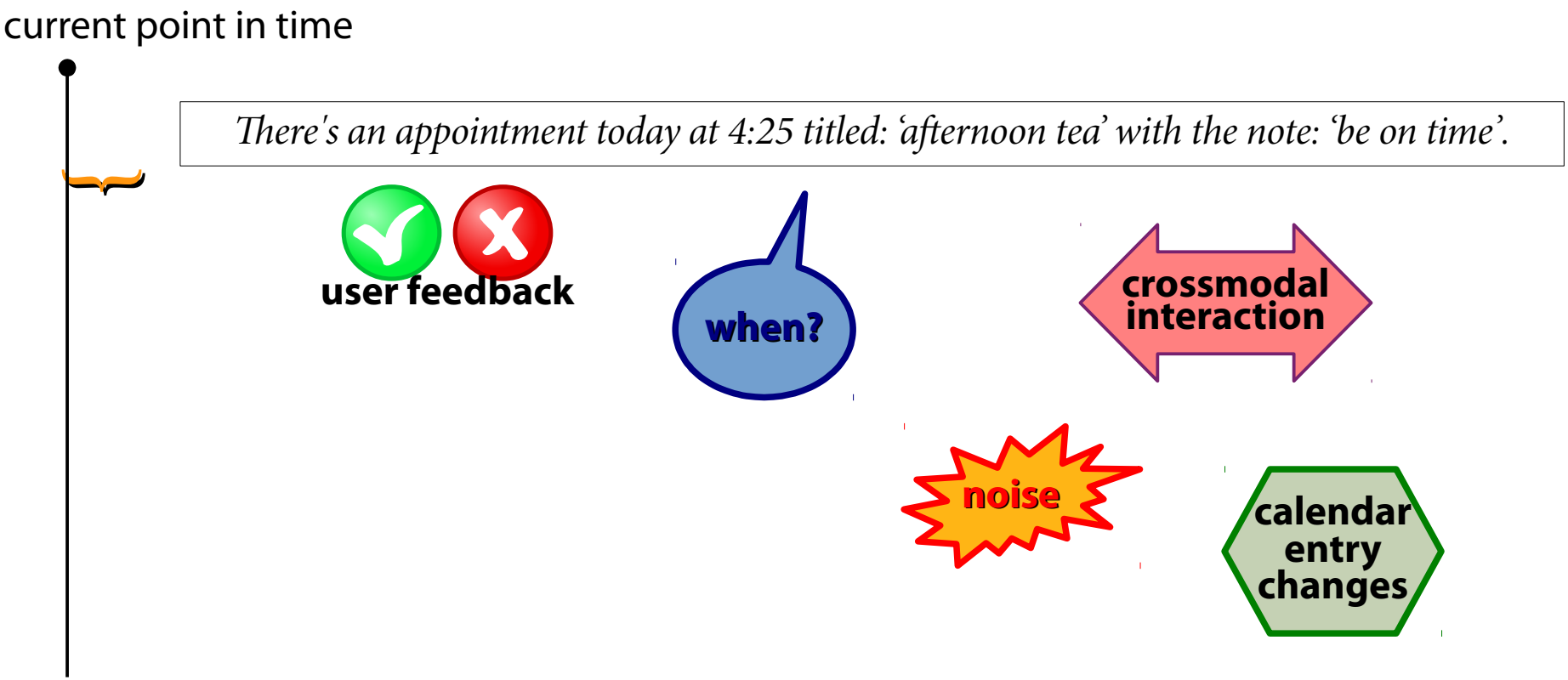
Baumann (2013)

Speech Output in Typical Systems



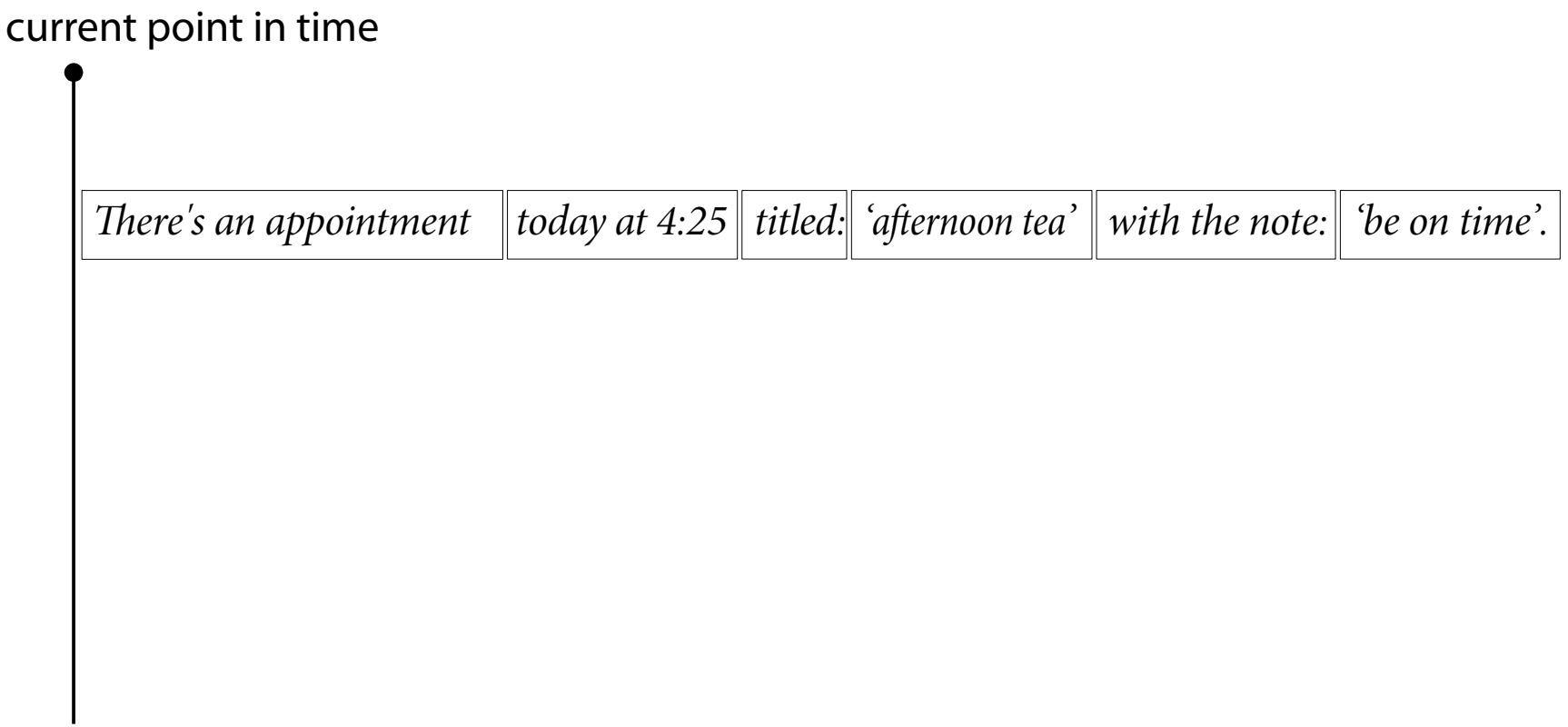
- full utterances are generated, synthesized and delivered as a whole

Speech Output in Typical Systems



- inflexible: unable to change the ongoing utterance (neither the content nor the delivery parameters)
  - no way to react to the listener or the environment

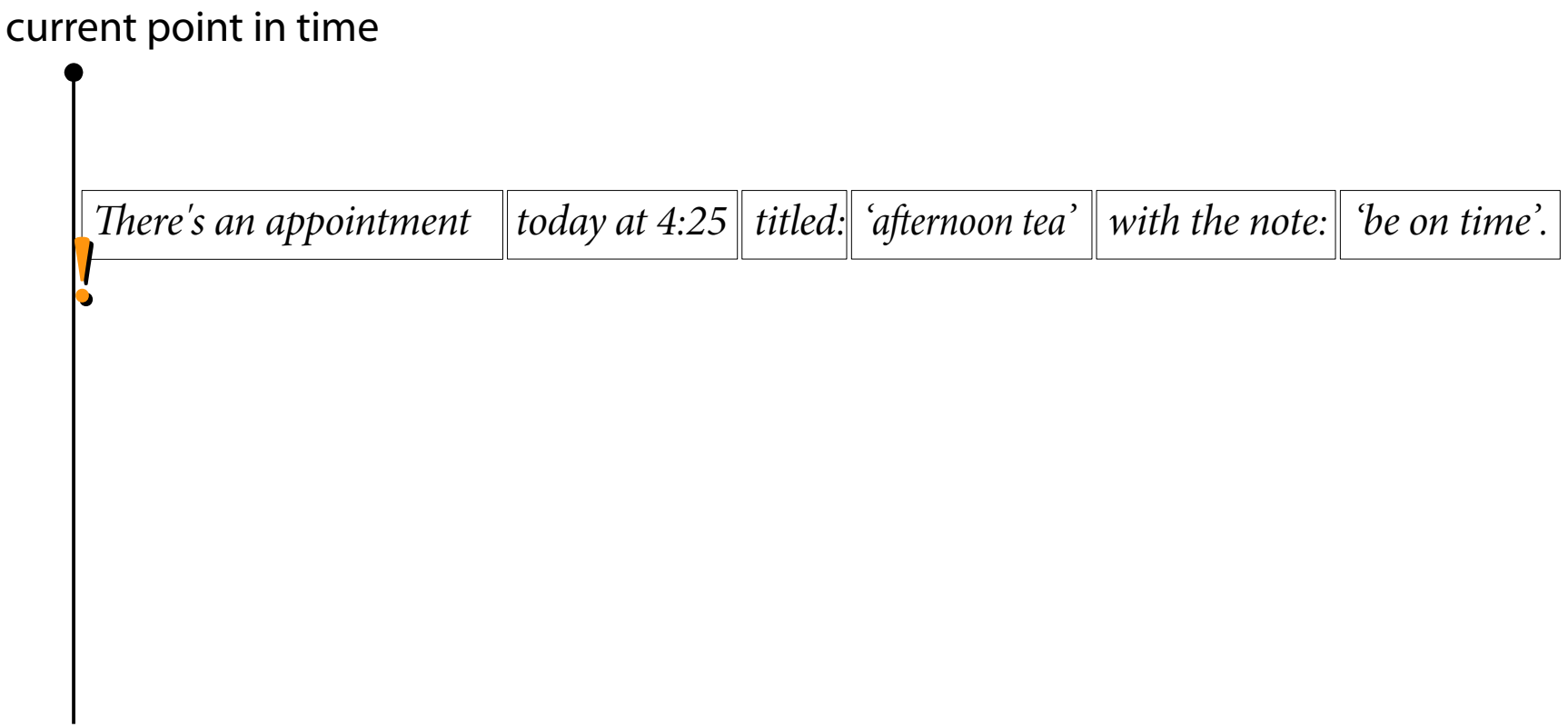
Potentially Better: Incremental Speech Output



- generate, synthesize and deliver the utterance in smaller *chunks*

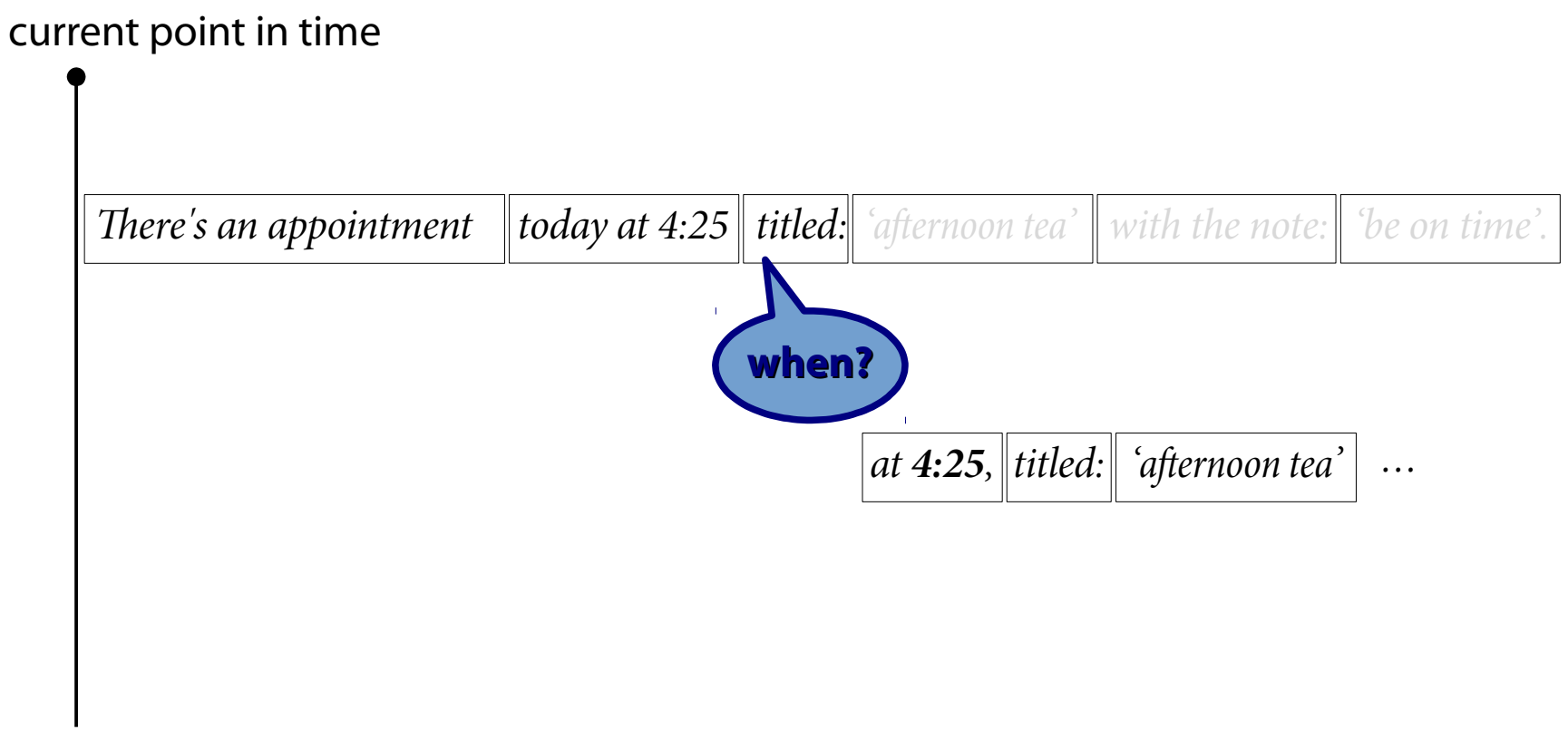
  → but (re)compute prosody with all the context available

Potentially Better: Incremental Speech Output



- less utterance-initial processing → faster onset

Potentially Better: Incremental Speech Output



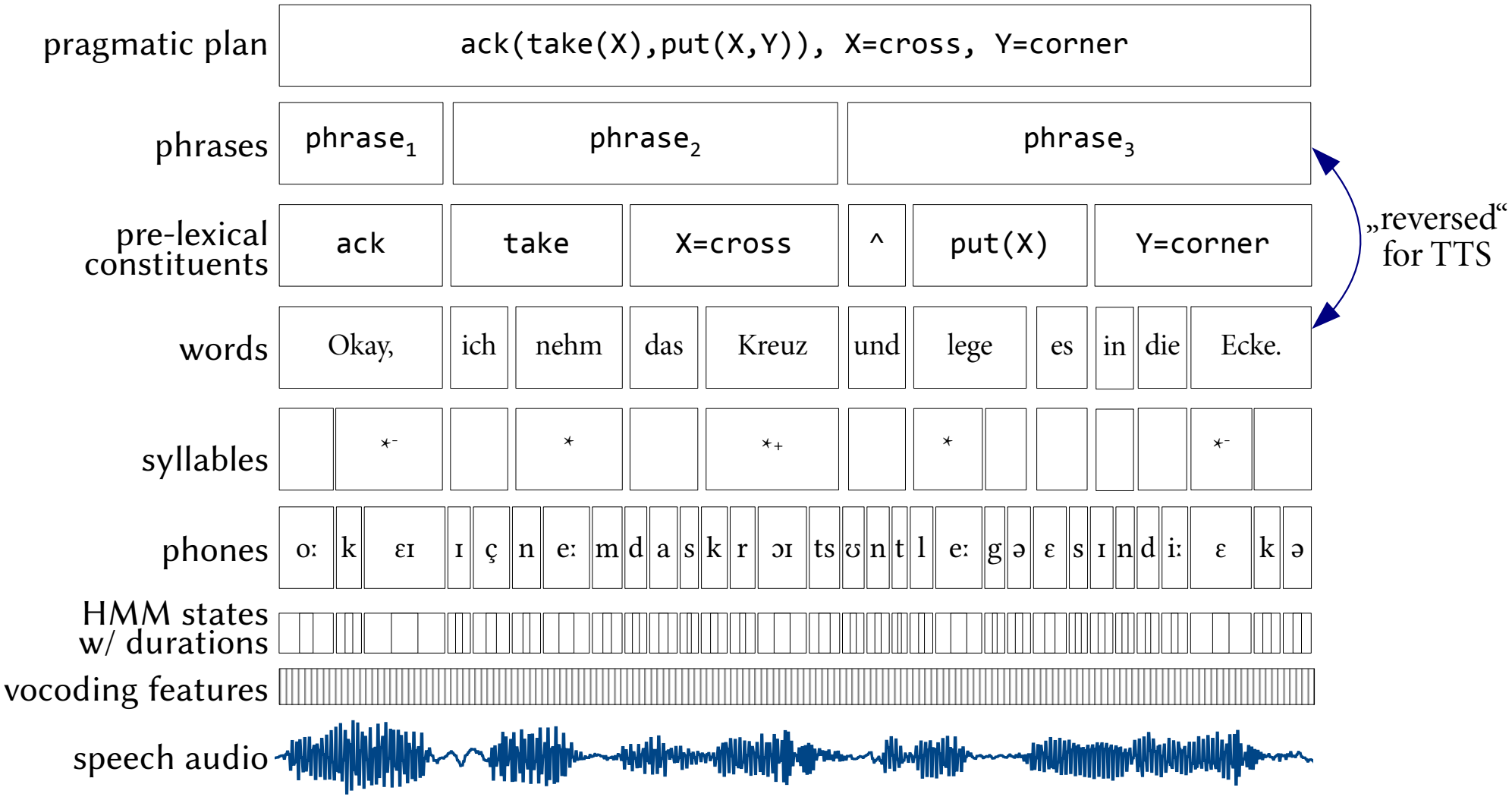
- incremental output may take *changes* into account
- react and adapt to user feedback / requests / noise

Example

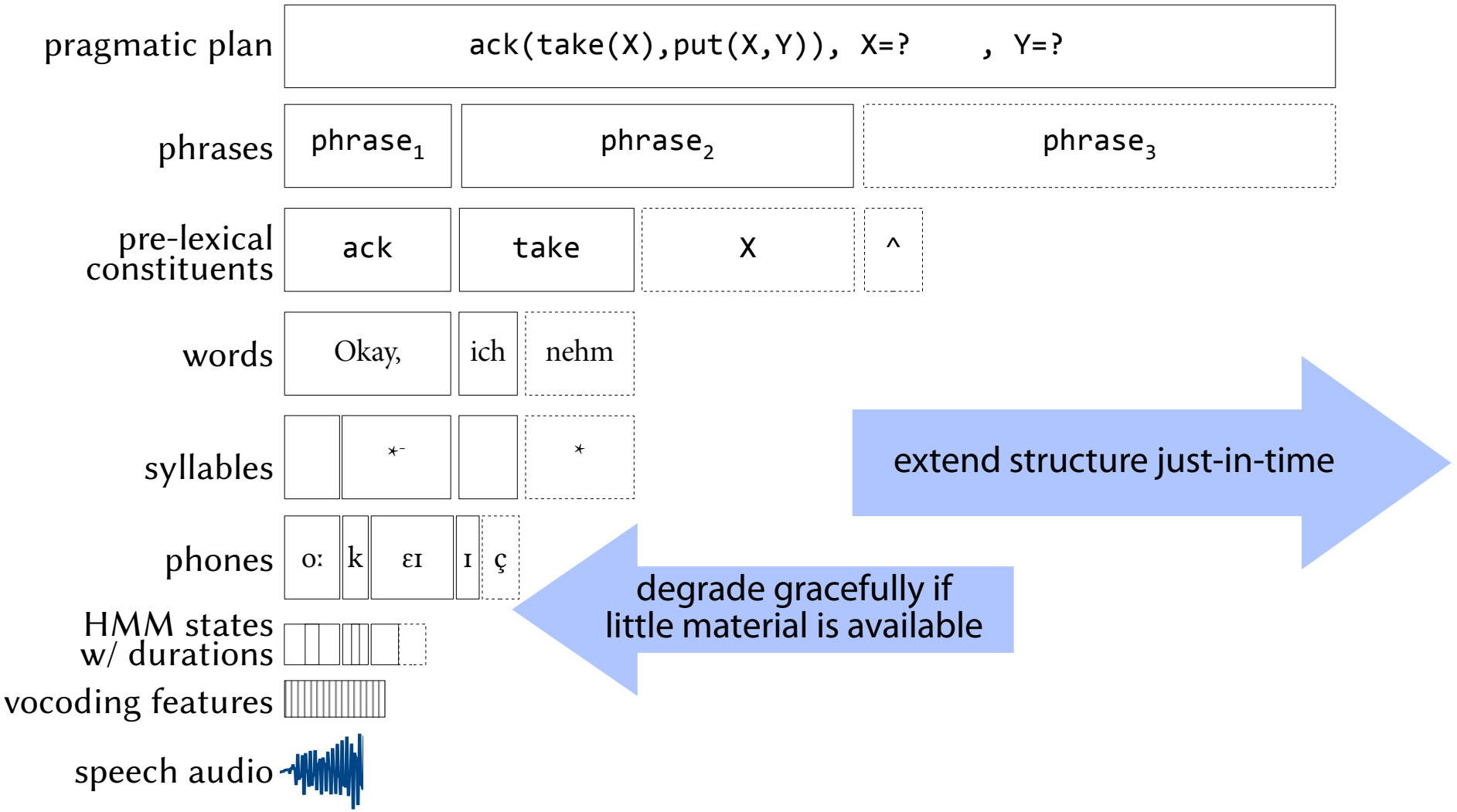

Umw
Auto aus Querstr. *Warnen?*
Gefahr außer Reichweite
Golf biegt auch ab!
Auto-sensorik
Fuß vom Gas
Kupplung
Bremse
Bremse gelöst
Nutzer-tracking
Fahrer schaut rechts
Fahrer schaut gradeaus
Fahrer schaut rechts (erwartbar)
Biegen ~~Sie rechts in 60 Metern ab.~~
~~Achtung~~ dem Golf folgend ab
System
Warnung vorbereiten: lauter werden, Text ändern
doch nicht, Text wie gehabt.
will der etwa hier schon abbiegen?
Korrigieren: anderen Bezugspunkt geben!

Incremental speech synthesis architecture

Standard Speech Generation and Synthesis (HMM-based)

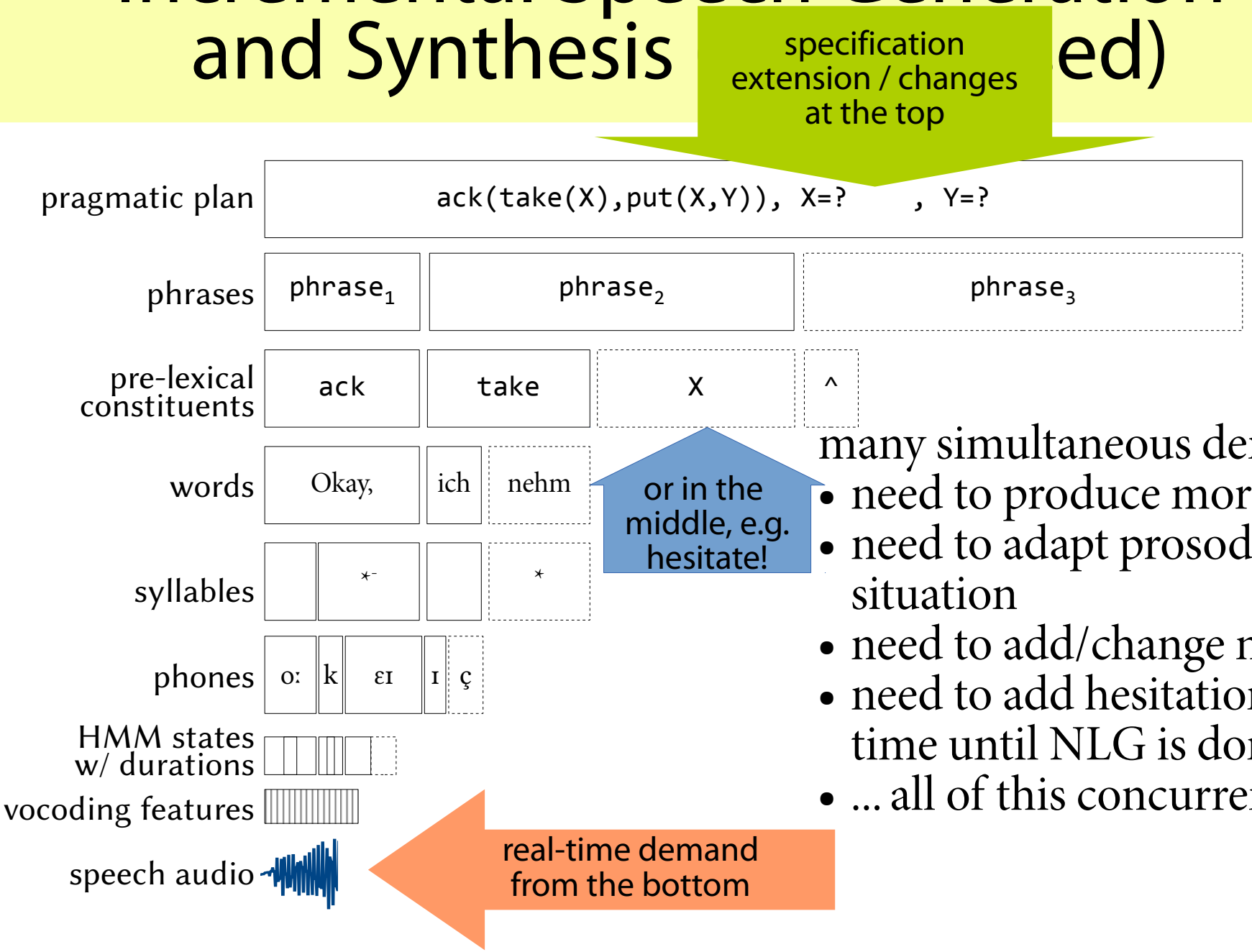


Incremental Speech Generation and Synthesis (HMM-based)



Baumann and Schlangen (2012a,b)
Baumann and Schlangen (2013a,b)
Baumann (2014a,b)

Incremental Speech Generation and Synthesis ed)

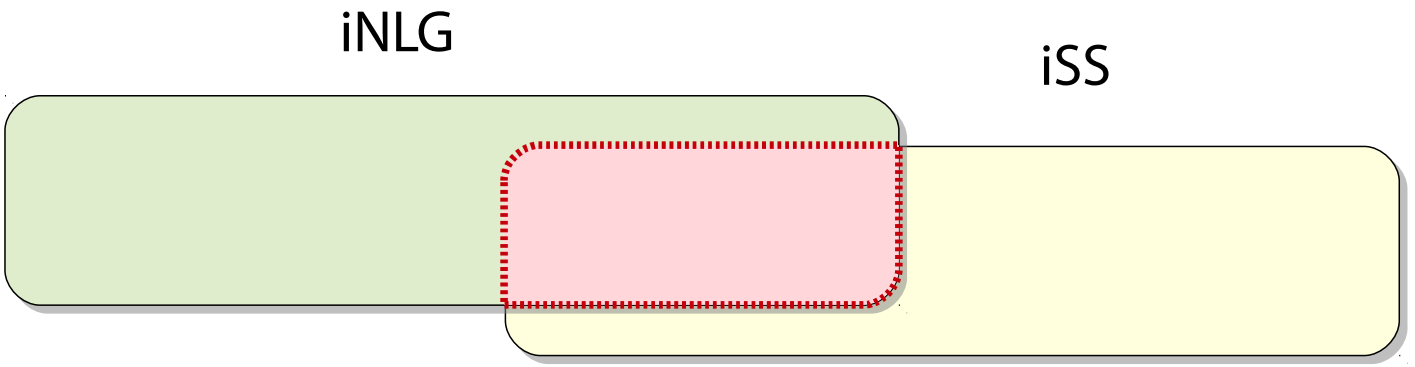


many simultaneous demands:

- need to produce more speech
- need to adapt prosody to situation
- need to add/change material
- need to add hesitation to span time until NLG is done
- ... all of this concurrently

Extending structure just-in-time

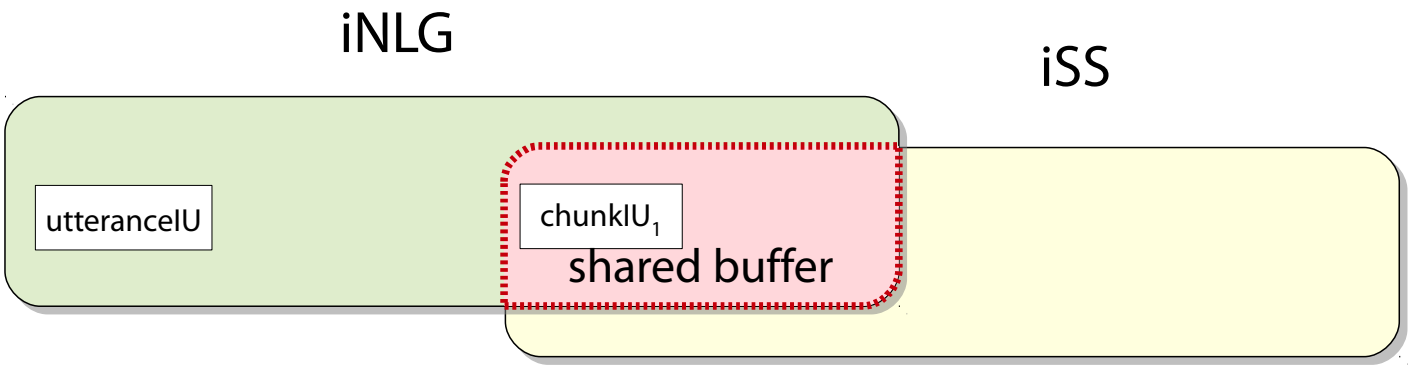
- split up into two (generic) processors:
  - natural language generation (iNLG)
  - speech synthesis (iSS)



- keep modularity as strong as possible

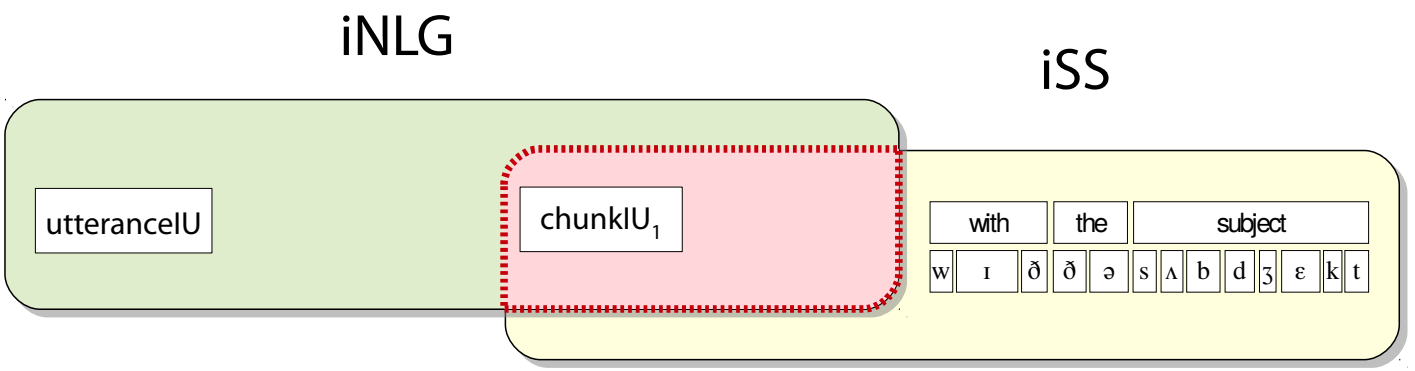
Incremental Speech Output: Overview

- starting with an utterance description
- iNLG splits the utterance in chunks and outputs one chunk to the buffer that is shared with iSS



Incremental Speech Output: Overview

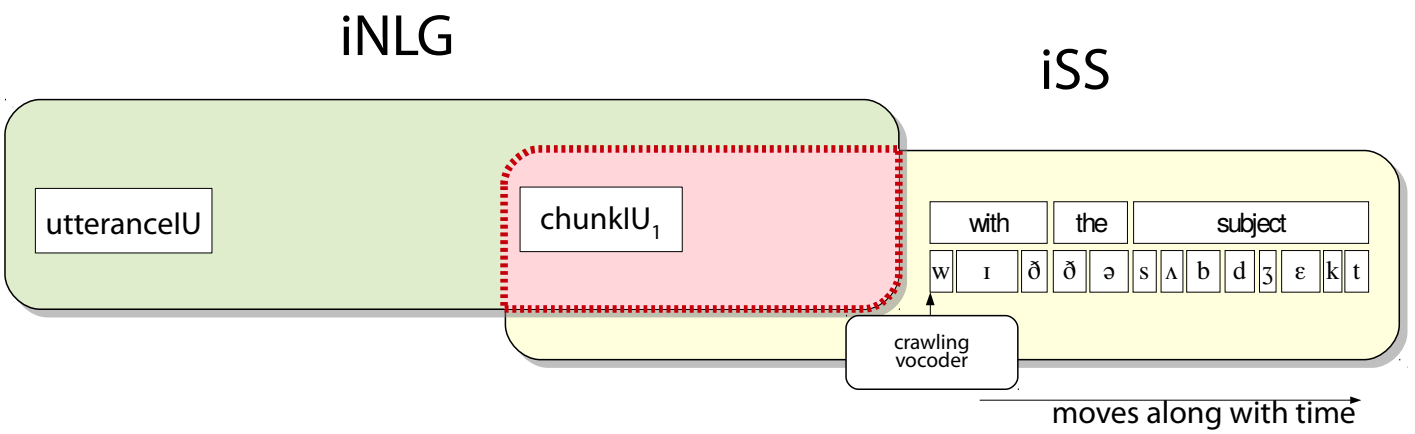
- iSS processes chunk to produce phonemes



linguistic processing via MaryTTS (Schröder & Trouvain, 2003)

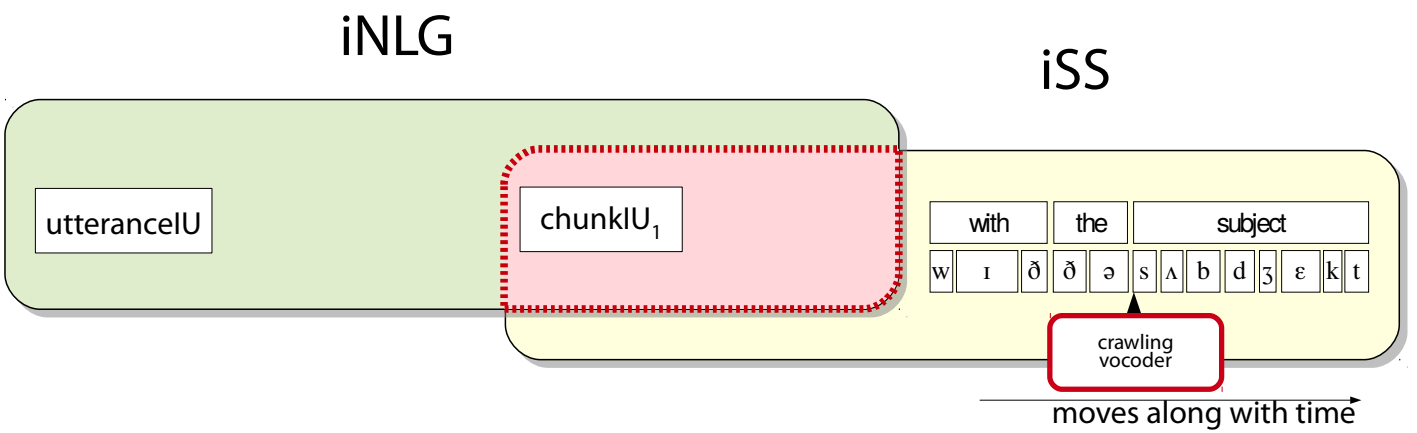
Incremental Speech Output: Overview

- iSS processes chunk and
- synthesizes *just-in-time*
  (only with enough look-ahead to keep all buffers full)



Incremental Speech Output: Overview

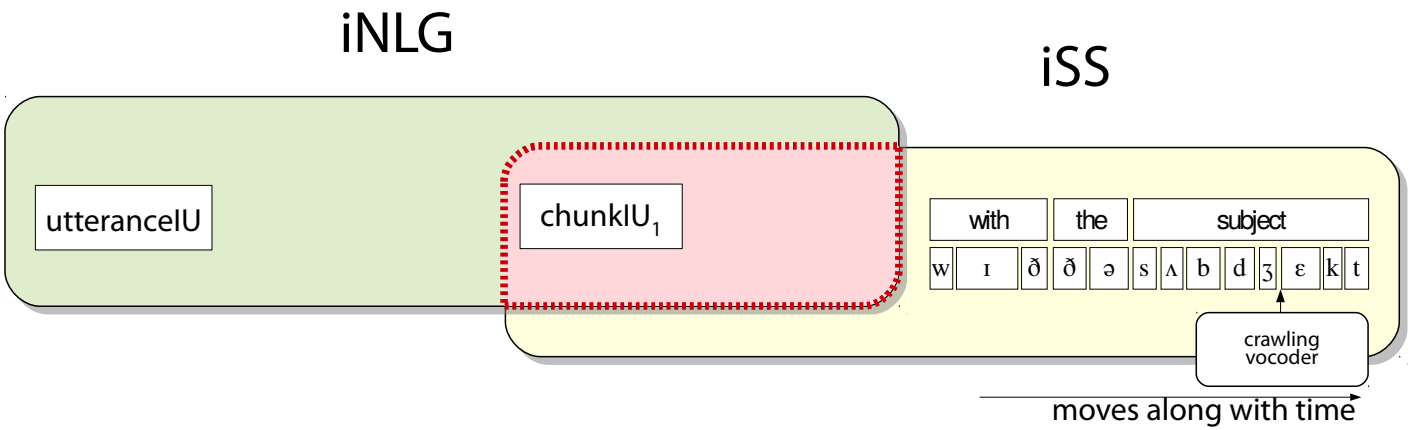
- using a *crawling vocoder* that performs HMM optimization and vocoding in real-time



(largely based on MaryTTS code; see also Dutoit et al., 2011)

Incremental Speech Output: Overview

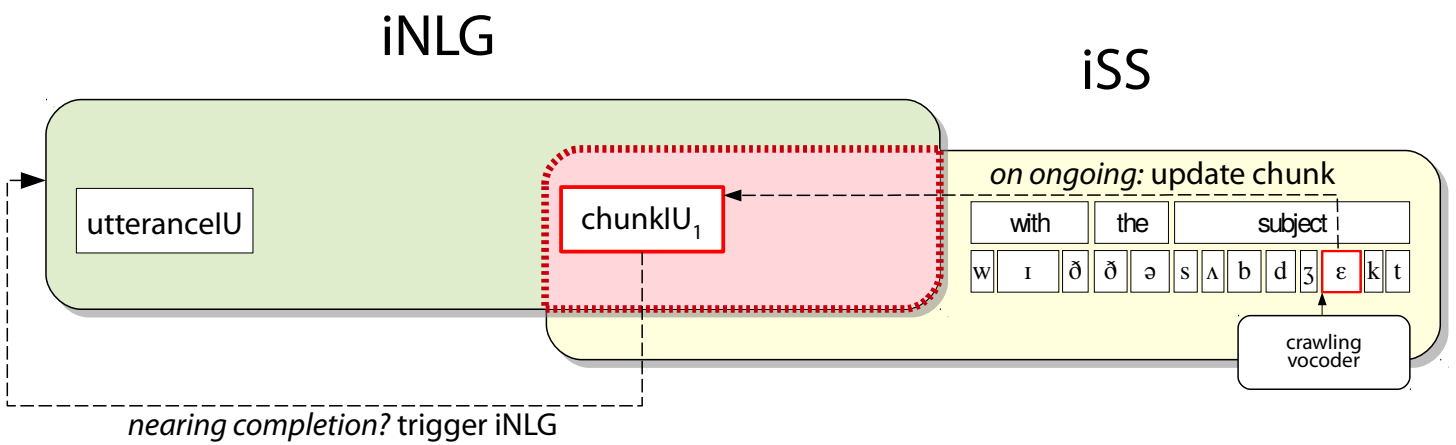
- using a *crawling vocoder* that performs HMM optimization and vocoding in real-time
- when nearing the end of the current chunk …



Incremental Speech Output: Overview

- update-messages are sent from phonemes to chunk to iNLG

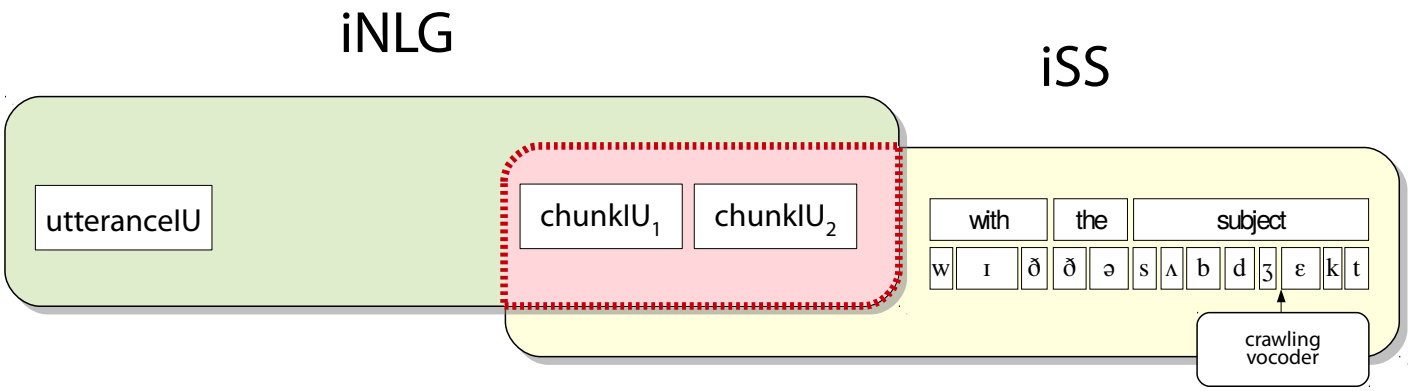
  (this is a generic update mechanism in INPROTK)



- update trigger placement determines (minimal) lookahead

Incremental Speech Output: Overview

- and iNLG adds another chunkIU before synthesis runs out of speech
- it's integrated & appended to the ongoing synthesis



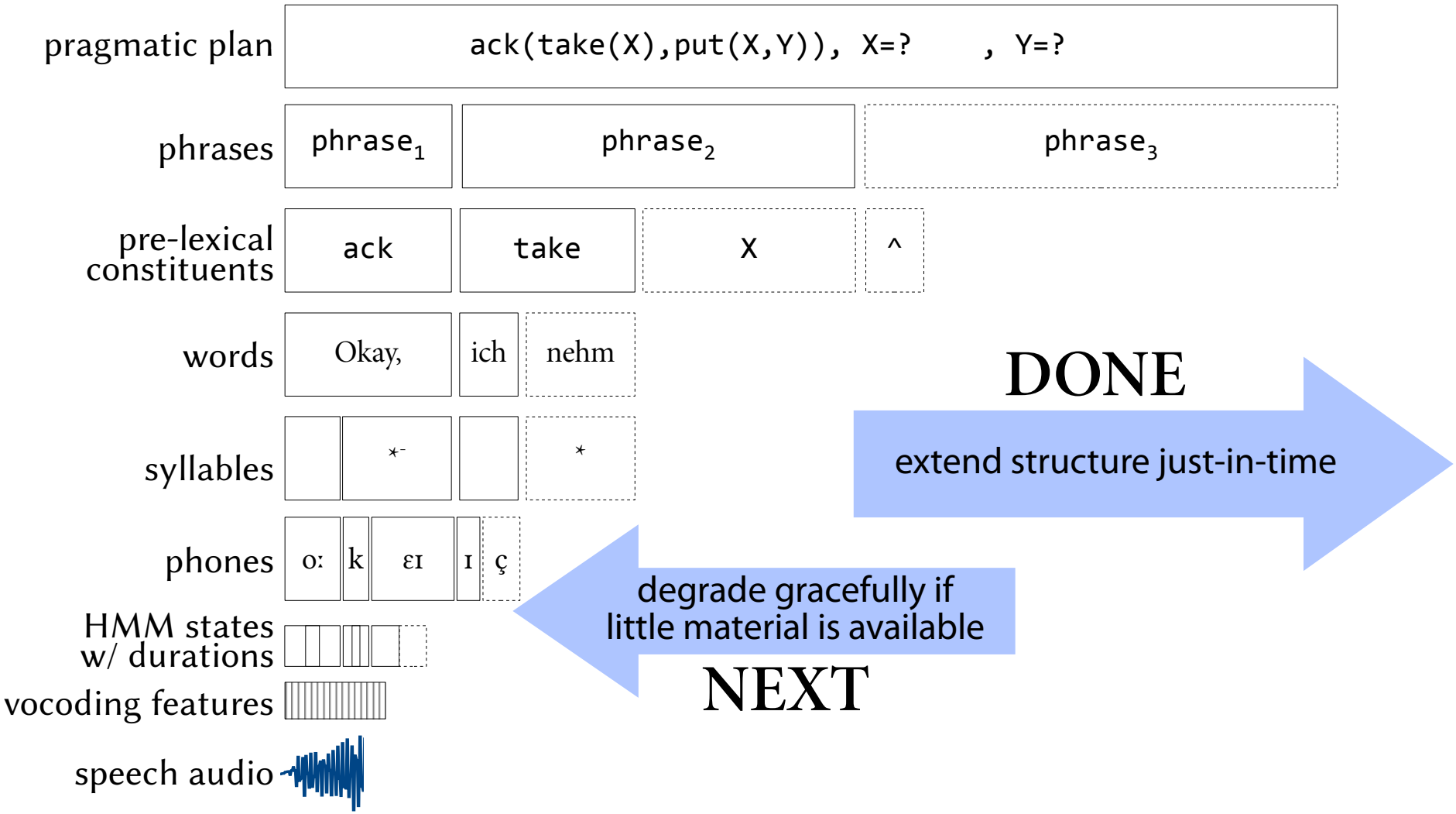
- the process repeats until all chunks are synthesized

Update mechanism

- updates notify higher-level processing
  that a processing step is required soon
- updates inform higher-level processing
  what can't be changed any more
  (where non-monotonicity is limited)

- WARNING: you may run into concurrency issues and race-conditions (probably with your code, certainly with mine!)

Incremental Speech Generation and Synthesis (HMM-based)



Baumann and Schlangen (2012a,b)
Baumann and Schlangen (2013a,b)
Baumann (2014a,b)

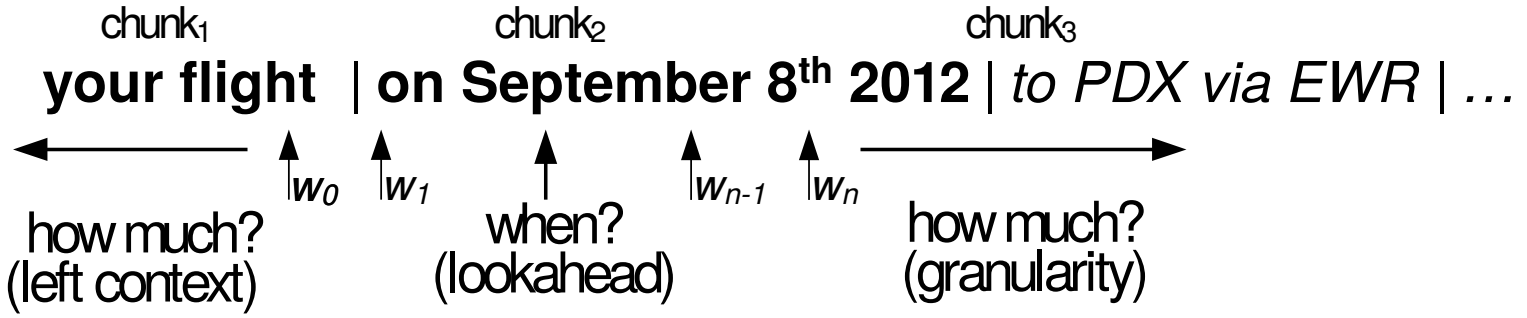
Prosody

- *the* non-local phenomenon in speech synthesis
  - other steps are very local; in particular: HSMM synthesis needs just 2 phonemes of context (Dutoit et al. 2011)
- we typically require the full sentence to compute the overall sentence intonation/melody
  - but can we get away with less than full sentences?
  - what's the degradation?
  - with how little can we get away?
- of course, more context will help more, but what about the interaction abilities that we gain from limiting context?

In-vitro evaluation: lookahead vs. prosodic quality

(a) for symbolic prosody processing (ToBI-like)
(b) sub-symbolic prosody processing (contour generation)

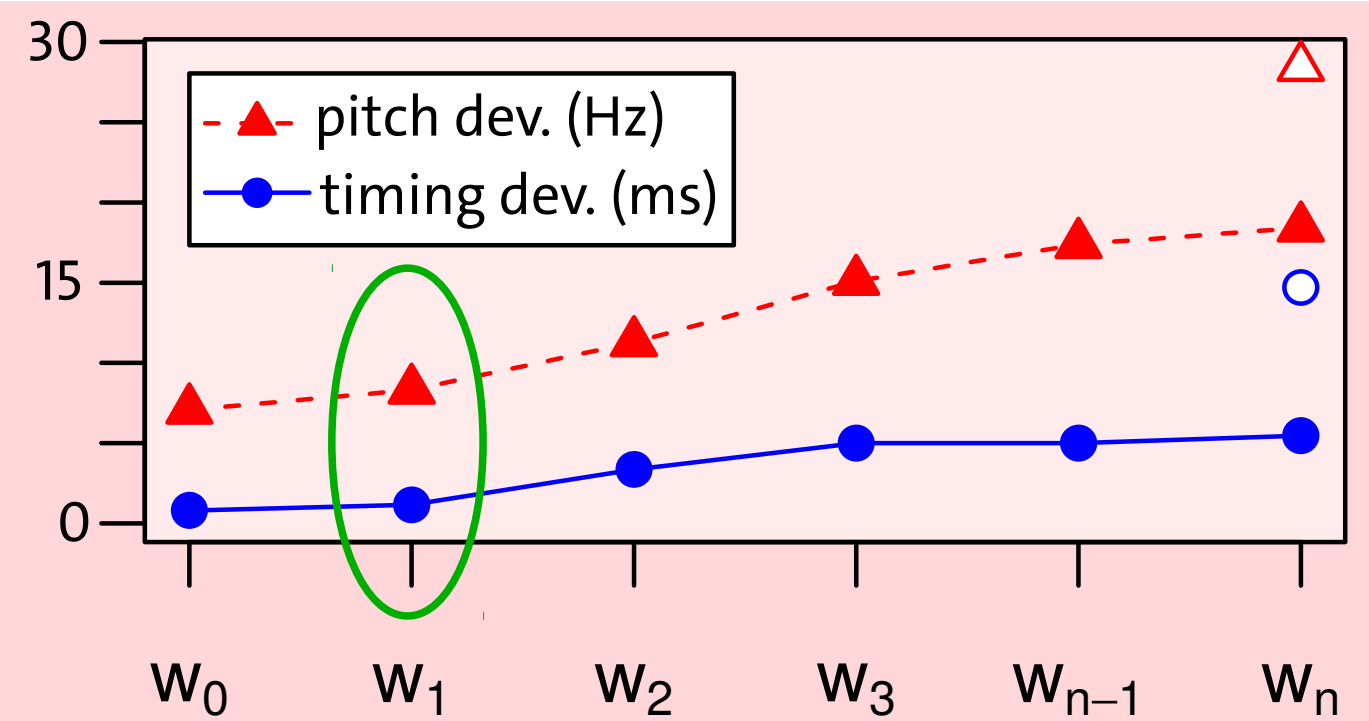
Design Space for Incremental Prosody



- phrases (as produced by a NLG component) may form a reasonable chunk-size for prosodic processing
  - NLG doesn't produce anything that's smaller anyway
- when we add input at w_t, we can change prosody for what's after w_t, but not before
  - the smaller t, the smaller the influence on prosody
  - the smaller t, the less incremental the synthesis

Evaluation

- we focus on pitch and duration error (RMSE) relative to non-incremental baseline

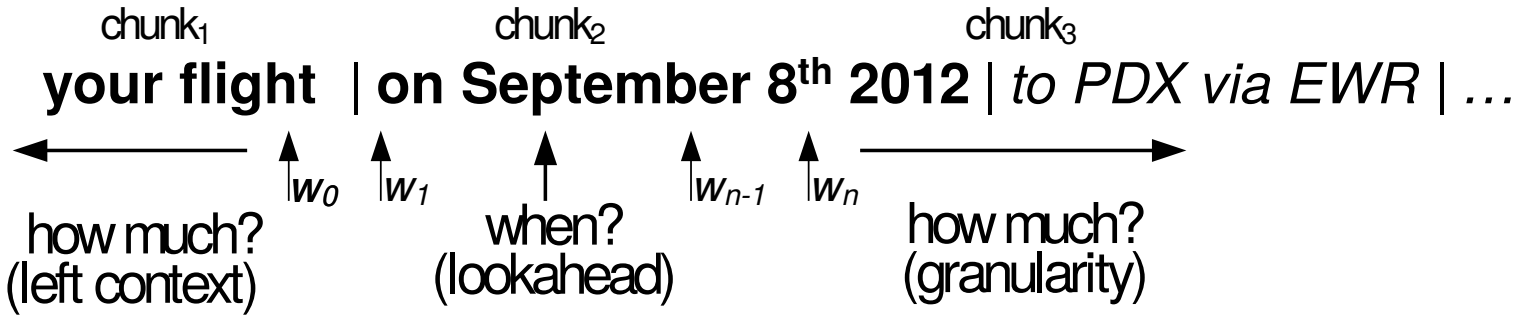


- add next phrase at end of current phrase's first word
- not very incremental, in particular: very coarse granularity

Trade-off

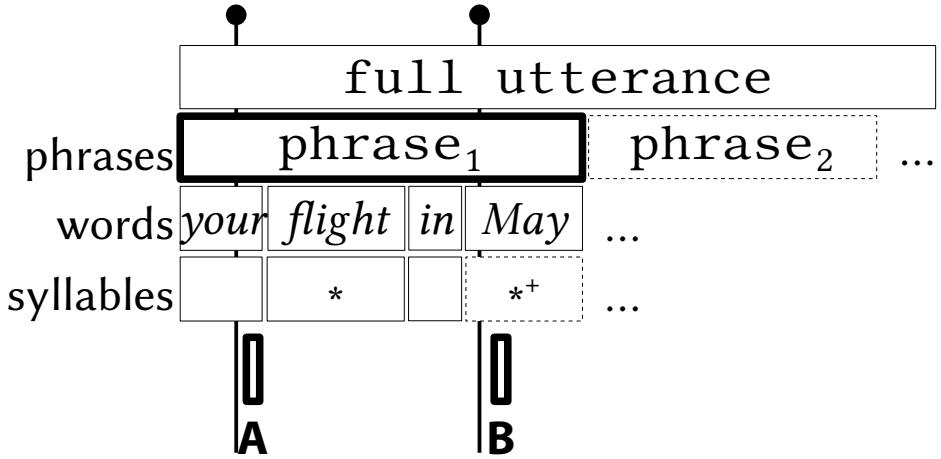
- more context, better prosodic quality
- more context, less incremental / timely changes
- (for application needs and measure *in vivo*)

- so far: we need (almost) two phrases of lookahead+granularity – can we do better?



More fine-grained processing

- try to reduce/mix granularity:
  - a synthesizer wants words phrase-by-phrase to answer phrase-level questions (such as „is this a question or a statement?“)
  - such phrase-based information may be more likely available at different stages into the phrase



phrase features likely
A: not available, **B:** available

  - are features important for the full phrase, or more important towards the end of the phrase?

Experimental setting

- limiting feature use to
  - everything that's in the past
  - 2 next phones and next syllable but only if they are part of the current word
  - word-level information up to the current word
  - phrase-level information only for the phrase-final word
  - sentence-level information only for the sentence-final word
- analyze state-selection for HSMM synthesis
  - in combination with full symbolic prosody
  - or just with previous w_{n-1} setting

details in Baumann, 2014 (Interspeech)

Experimental Results

- you don't really need all the forward-looking features
- phrase- and sentence-level information on the last word is sufficient
  - nicely corresponds to the fact that speech itself is an incremental phenomenon; human speakers speak incrementally
    - they can incorporate late changes (without sounding unrealistic)
    - we mark finality at the end of phrases/sentences

      (thus, linguistic insight would have come to the same conclusion as my tedious experiments...)

General take-away messages

- understand your underlying problem: what input is really needed, should be sufficient, might be dispensable?

  (e.g.: word+finality information almost as good as full-phrase)

- decompose your input into smaller units

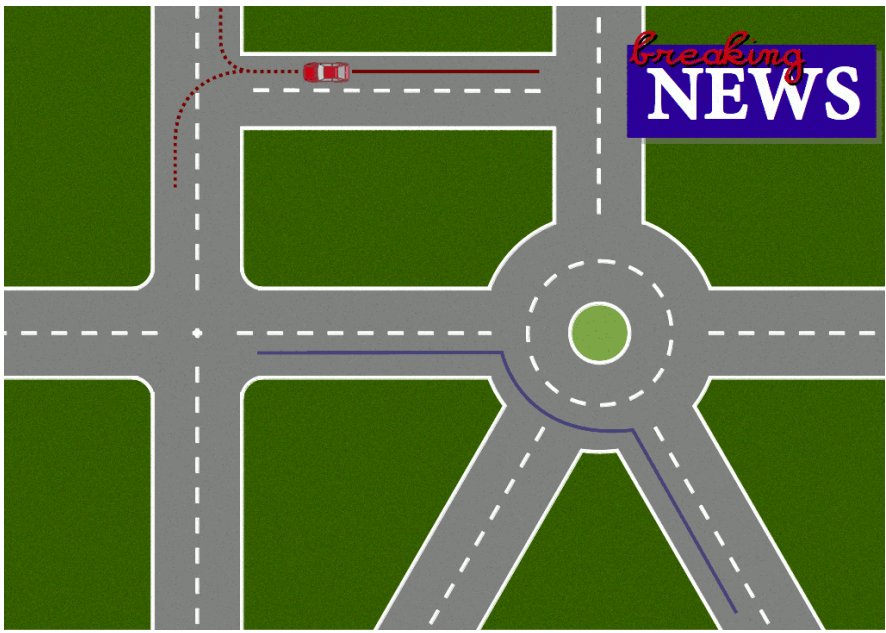
  (word and finality must be handled separately)

- devise *in-vitro* evaluation settings that make sense *in-vivo*

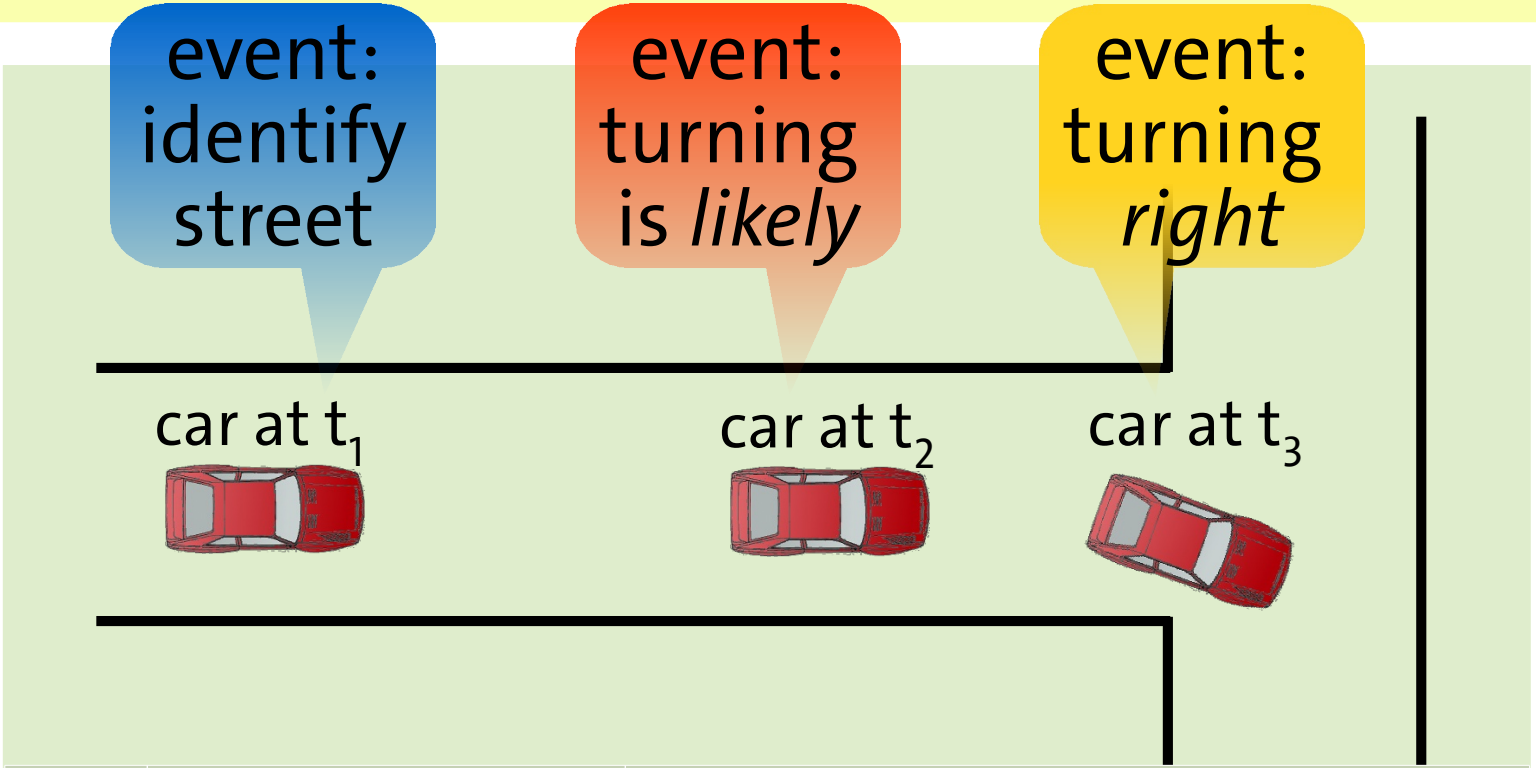
In-vivo evaluation

Example: The CarChase domain

- system comments on events in the scene (car's motion)
- high event rate → impossible to speak isolated utterances
  - combine events into complex utterances (using incremental speech synthesis)
  - skip or abort event notifications in favour of more important information (baseline behaviour)
- simplification of similar real-world scenarios



Taking expectations into account



time	event description	ongoing utterance (spoken part in **bold**)
t_1	car on Main Street	**The** car drives along Main Street.
t_2	car will have to turn	... **Main Str**eet and then turns ‹hes›
t_3	car turns right	... **Main Street and the**n turns right.

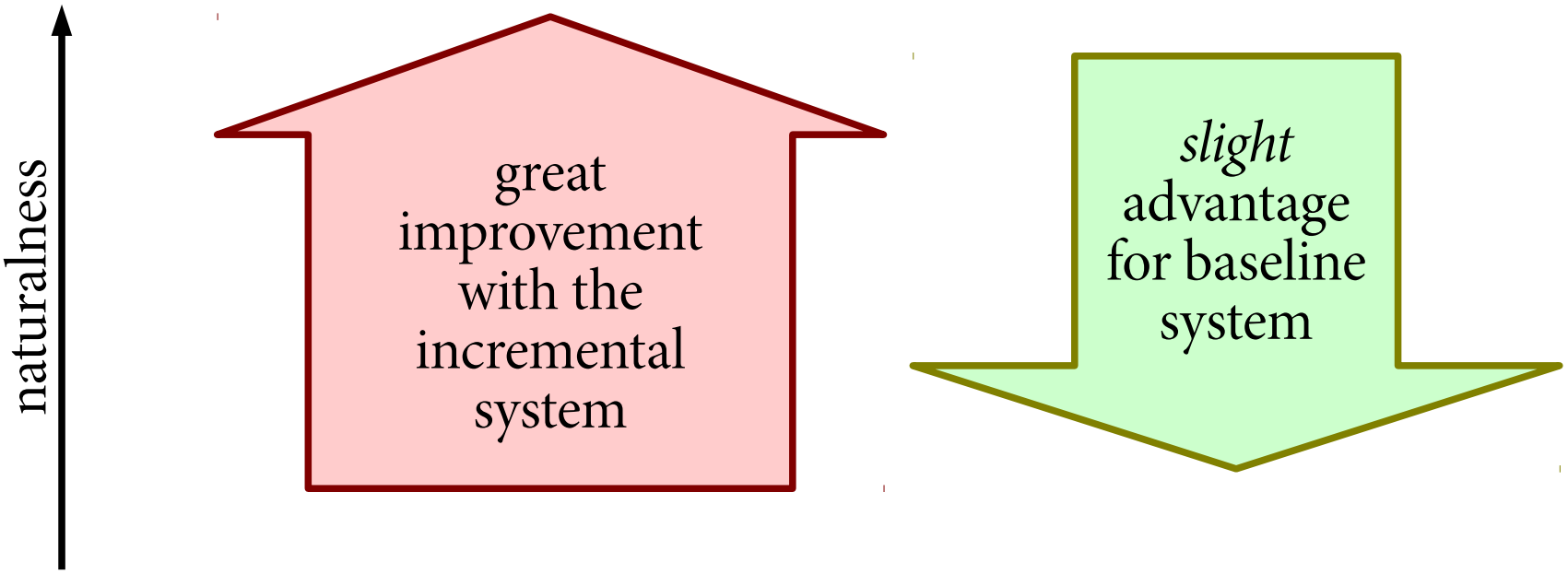
more details on interaction strategy in Baumann&Schlangen, SigDial 2013.

Experiment

- incremental system vs. baseline system
- 9 settings in the CarChase domain
- 9 subjects were asked to rate (5-point Likert)
  - naturalness of verbalization (to capture interactional adequacy)
  - naturalness of *pronunciation* (to capture synthesis quality)
- results in 81 paired samples

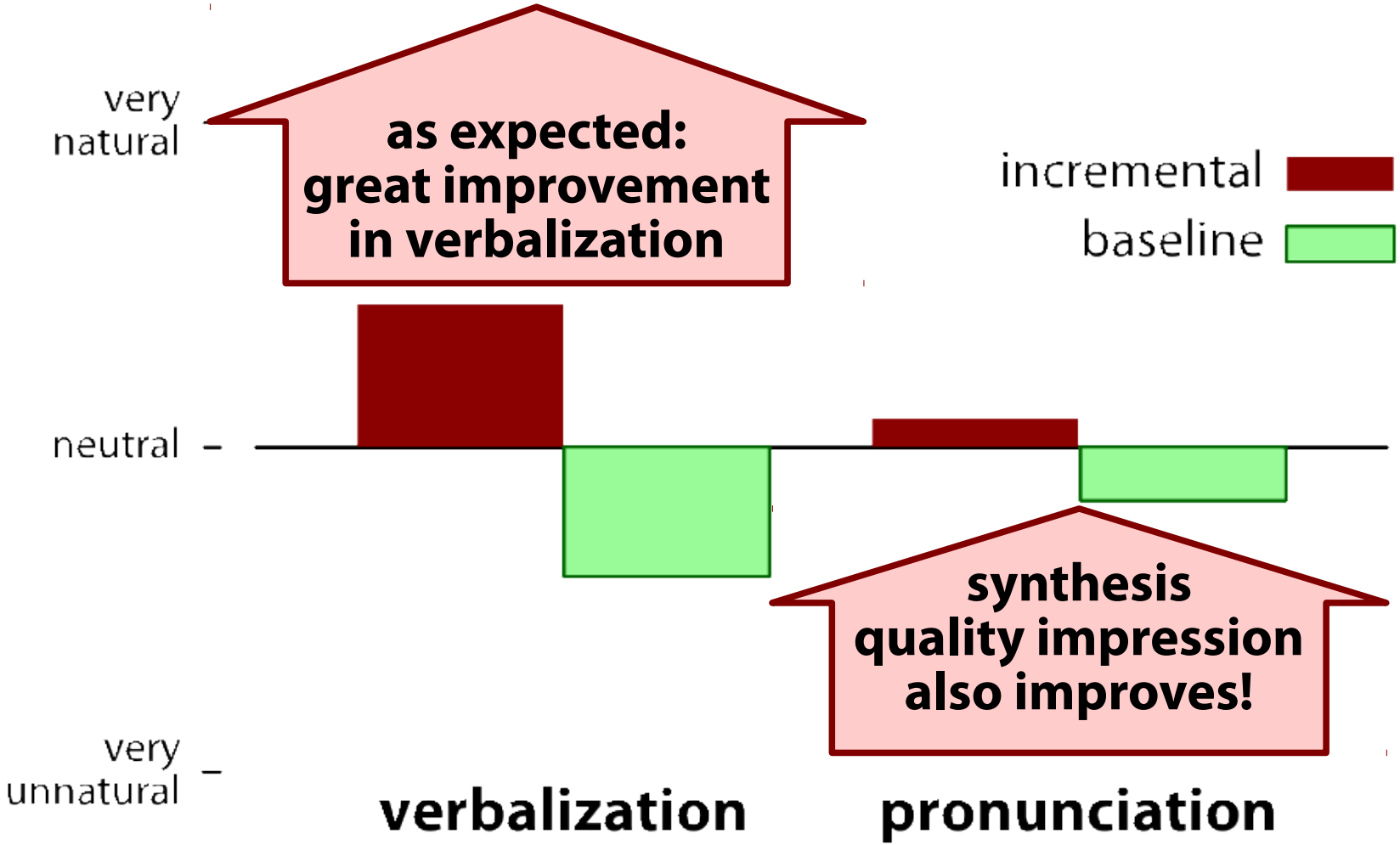
Expected results

- we were hoping for a good trade-off:



→ write paper: „Trade-off between incrementality of behaviour and speech synthesis quality“

Actual results



Pronunciation ratings

- Incremental processing cannot have systematically improved synthesis quality

- but:
  naïve listeners do not distinguish between interaction and synthesis quality (Pearson's r = .537)
- verbalization/wording adequacy by far outweighs pronunciation/synthesis quality

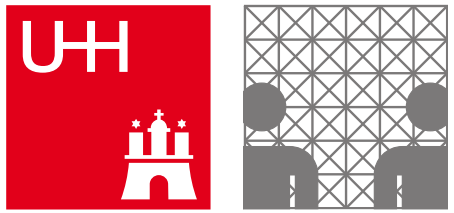
Conclusion

- Incremental speech synthesis sounds OK (similarly well as non-incremental speech synthesis)
  - quality/lookahead-tradeoff has reasonable operating points
  - mixed-granularity offers best solutions

- Prosody is the bottleneck

  better integration → partial structures → better prosody
  - (light) CTS instead of TTS

- inc. speech synthesis enables speech output that is rated as more natural than standard, non-incremental speech output



Thank you.

{baumann,koehn}@informatik.uni-hamburg.de
get the code at inprotk.sf.net.

page intentionally left blank

Desired Learning Outcomes

- students know solutions of how to approach a large and multi-facetted „incrementalization“ problem: partitioning, reset-incremental processing, processing vs. programming overhead
- students know how to re-arrange complex processing to suit incrementality; they understand the basics of incrementally producing speech output
- students understand that incremental processing involves concurrency issues, in particular when generating output
- students can assess quality tradeoffs in interaction, e.g. between responsiveness and speech quality