Enumerative Data Types with Constraints

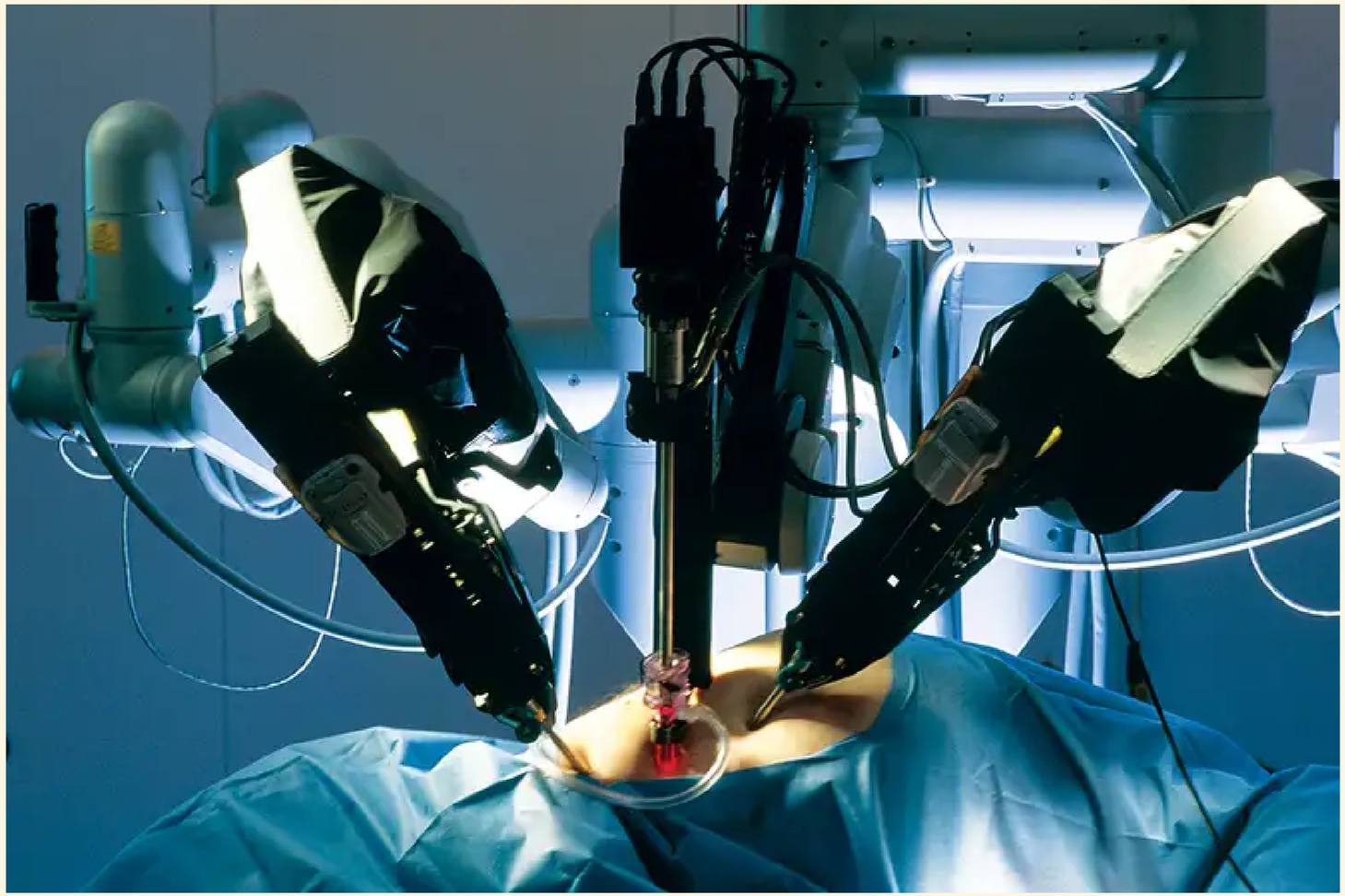
FMCAD 2022

2022-10-19

Andrew T. Walter[1], David Greve[2], Panagiotis Manolios[1]

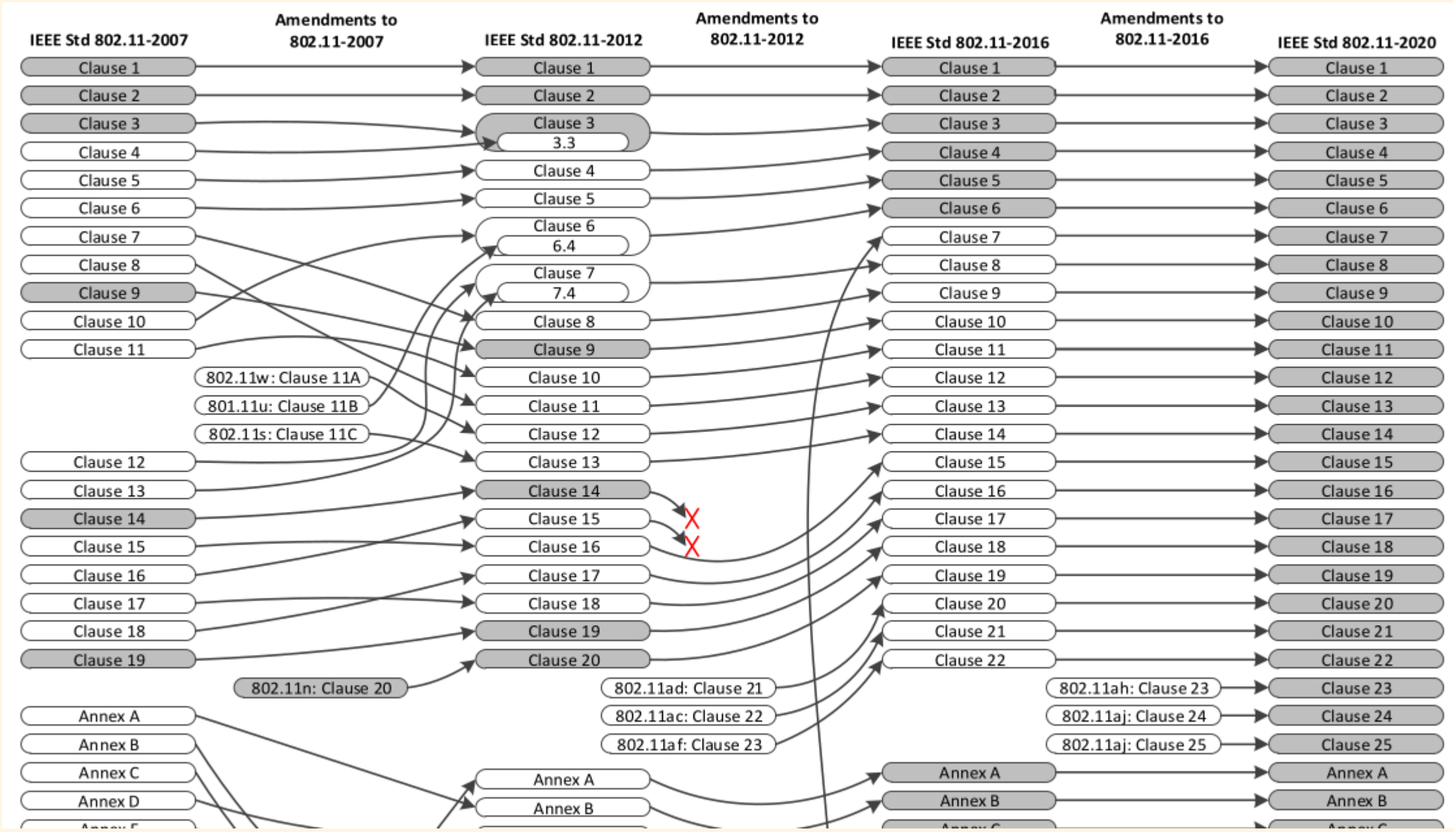
1. Khoury College of Computer Science, Northeastern University
2. Collins Aerospace

Analyzing a Robotic Surgeon



From https://www.newscientist.com/article/mg24232342-300-robotic-surgery-is-turning-out-to-be-an-expensive-fad/

WI-FI PACKET GENERATION IS HARD



From "IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks--Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," in IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) , vol., no., pp.1-4379, 26 Feb. 2021, doi: 10.1109/IEEESTD.2021.9363693.

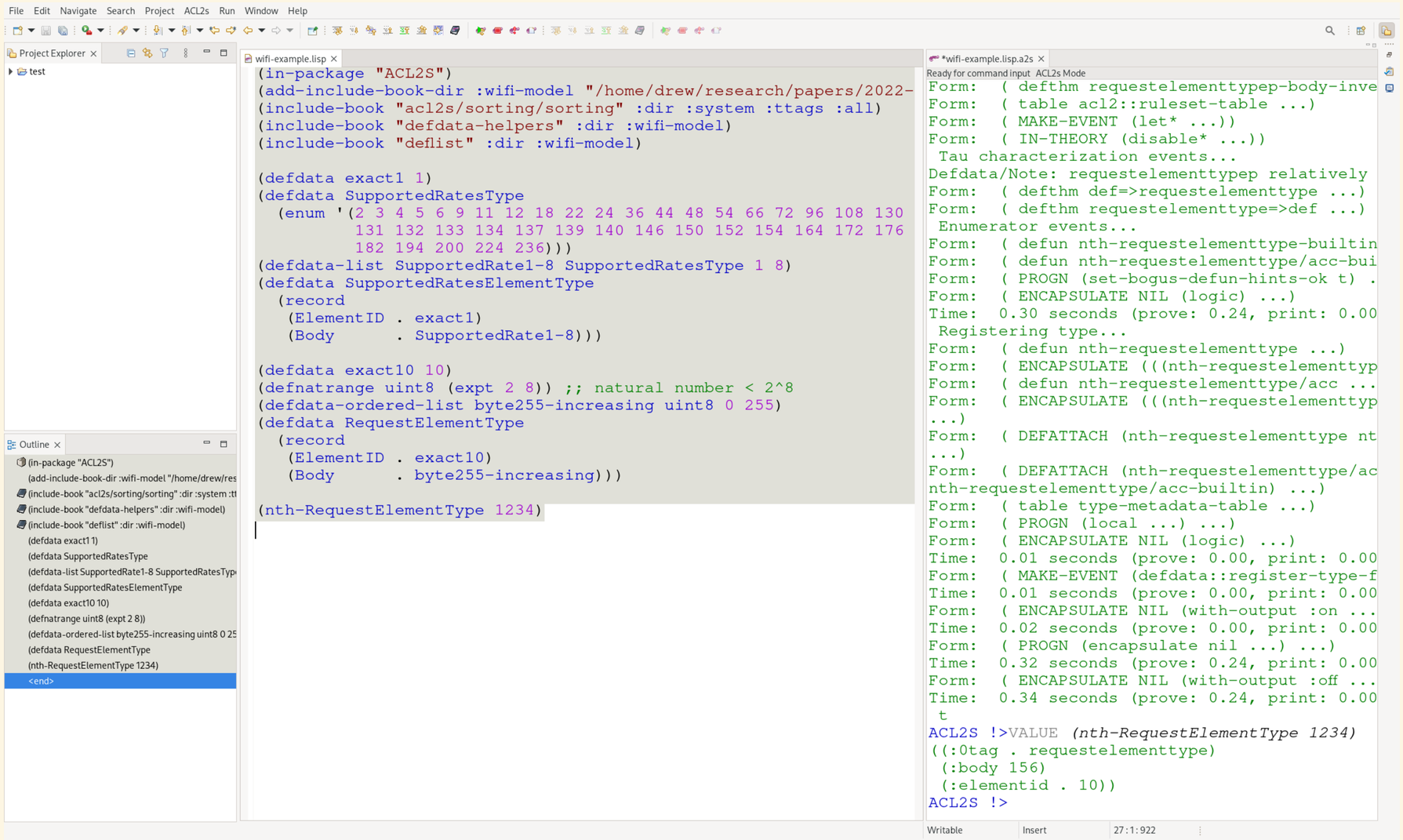
SMTLIB

```
(declare-datatype SupportedRate ((|2|) (|3|) ... (|224|) (|236|)))
(declare-datatype SupportedRatesElement
                  ((SupportedRatesElement
                    (SupportedRatesElement/Id Int)
                    (SupportedRatesElement/Body (Seq SupportedRate)))))
(declare-const sre SupportedRatesElement)
;; Note that we need to specify these as assertions rather than as
;; part of the type definition for SupportedRatesElement
(assert (and (>= (seq.len (SupportedRatesElement/Body sre)) 1)
             (<= (seq.len (SupportedRatesElement/Body sre)) 8)
             (= (SupportedRatesElement/Id sre) 1)))
```

SMTLIB

```
(declare-datatype RequestElement                    ;; Over 2800 LoC!
                  ((RequestElement                  ;; See repo for model.
                    (RequestElement/Id Int)
                    (RequestElement/Length Int)
                    (RequestElement/Body (Seq (_ BitVec 8)))))
(declare-const req RequestElement)
;; Note that we need to specify these as assertions rather than as
;; part of the type definition for RequestElement
(assert (and (= (RequestElement/Id req) 10)
             (= (RequestElement/Length req)
                (seq.len (RequestElement/Body req)))
             (<= (seq.len (RequestElement/Body req)) 255)
             (or (<= (RequestElement/Length req) 1)
                 (bvult (seq.nth (RequestElement/Body req) 0)
                        (seq.nth (RequestElement/Body req) 1))
             ... ;; 252 occurrences elided
             (or (<= (RequestElement/Length req) 254)
                 (bvult (seq.nth (RequestElement/Body req) 254)
                        (seq.nth (RequestElement/Body req) 255))))
```

ACL2S


File Edit Navigate Search Project ACL2s Run Window Help
Project Explorer
test
wifi-example.lisp
(in-package "ACL2S")
(add-include-book-dir :wifi-model "/home/drew/research/papers/2022-
(include-book "acl2s/sorting/sorting" :dir :system :ttags :all)
(include-book "defdata-helpers" :dir :wifi-model)
(include-book "deflist" :dir :wifi-model)
(defdata exact1 1)
(defdata SupportedRatesType
(enum '(2 3 4 5 6 9 11 12 18 22 24 36 44 48 54 66 72 96 108 130
131 132 133 134 137 139 140 146 150 152 154 164 172 176
182 194 200 224 236)))
(defdata-list SupportedRate1-8 SupportedRatesType 1 8)
(defdata SupportedRatesElementType
(record
(ElementID . exact1)
(Body . SupportedRate1-8)))
(defdata exact10 10)
(defnatrange uint8 (expt 2 8)) ;; natural number < 2^8
(defdata-ordered-list byte255-increasing uint8 0 255)
(defdata RequestElementType
(record
(ElementID . exact10)
(Body . byte255-increasing)))
(nth-RequestElementType 1234)
Outline
(in-package "ACL2S")
(add-include-book-dir :wifi-model "/home/drew/res
(include-book "acl2s/sorting/sorting" :dir :system :tt
(include-book "defdata-helpers" :dir :wifi-model)
(include-book "deflist" :dir :wifi-model)
(defdata exact1 1)
(defdata SupportedRatesType
(defdata-list SupportedRate1-8 SupportedRatesTyp
(defdata SupportedRatesElementType
(defdata exact10 10)
(defnatrange uint8 (expt 2 8))
(defdata-ordered-list byte255-increasing uint8 0 25
(defdata RequestElementType
(nth-RequestElementType 1234)
<end>
*wifi-example.lisp.a2s
Ready for command input ACL2s Mode
Form: (defthm requestelementtypep-body-inve
Form: (table acl2::ruleset-table ...)
Form: (MAKE-EVENT (let* ...))
Form: (IN-THEORY (disable* ...))
Tau characterization events...
Defdata/Note: requestelementtypep relatively
Form: (defthm def=>requestelementtype ...)
Form: (defthm requestelementtype=>def ...)
Enumerator events...
Form: (defun nth-requestelementtype-builtin
Form: (defun nth-requestelementtype/acc-bui
Form: (PROGN (set-bogus-defun-hints-ok t) .
Form: (ENCAPSULATE NIL (logic) ...)
Time: 0.30 seconds (prove: 0.24, print: 0.00
Registering type...
Form: (defun nth-requestelementtype ...)
Form: (ENCAPSULATE (((nth-requestelementtyp
Form: (defun nth-requestelementtype/acc ...
Form: (ENCAPSULATE (((nth-requestelementtyp
...)
Form: (DEFATTACH (nth-requestelementtype nt
...)
Form: (DEFATTACH (nth-requestelementtype/ac
nth-requestelementtype/acc-builtin) ...)
Form: (table type-metadata-table ...)
Form: (PROGN (local ...) ...)
Form: (ENCAPSULATE NIL (logic) ...)
Time: 0.01 seconds (prove: 0.00, print: 0.00
Form: (MAKE-EVENT (defdata::register-type-f
Time: 0.01 seconds (prove: 0.00, print: 0.00
Form: (ENCAPSULATE NIL (with-output :on ...
Time: 0.02 seconds (prove: 0.00, print: 0.00
Form: (PROGN (encapsulate nil ...) ...)
Time: 0.32 seconds (prove: 0.24, print: 0.00
Form: (ENCAPSULATE NIL (with-output :off ...
Time: 0.34 seconds (prove: 0.24, print: 0.00
t
ACL2S !>VALUE (nth-RequestElementType 1234)
((:0tag . requestelementtype)
(:body 156)
(:elementid . 10))
ACL2S !>
Writable
Insert
27 : 1 : 922

ACL2S MODEL

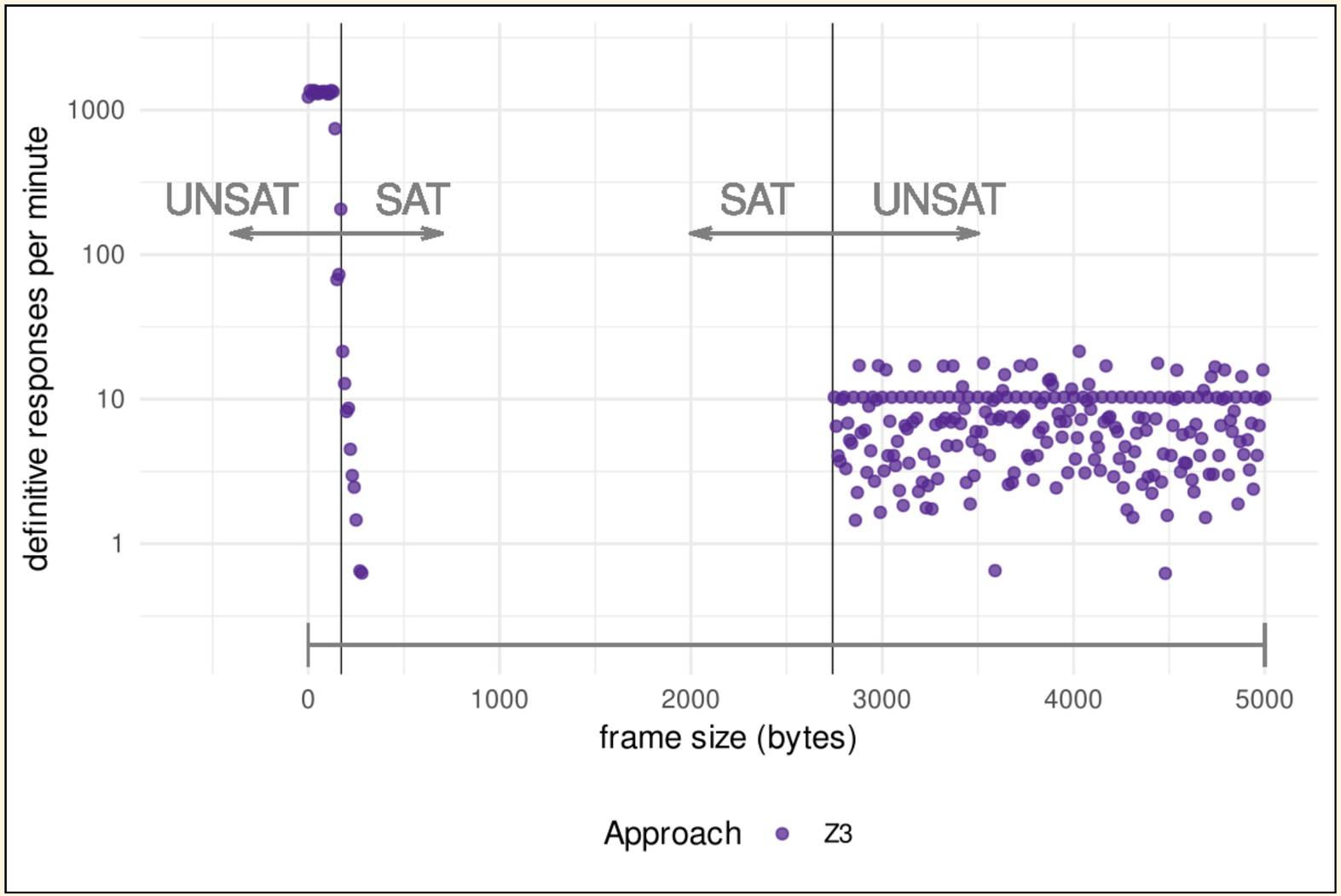
```
(defdata exact1 1)
(defdata SupportedRatesType (enum '(2 3 4 ... 200 224 236)))
;; This describes a list with length 1-8 inclusive with elements
;; that all have type supportedRatesType
(defdata-list SupportedRate1-8 SupportedRatesType 1 8)
(defdata SupportedRatesElementType
  (record
   (ElementID . exact1)
   (Body      . SupportedRate1-8)))
```

ACL2S MODEL

```
(defdata exact10 10) ;; 7 ACL2s lines vs 2k (extended) SMTLIB lines!
(defnatrange uint8 (expt 2 8)) ;; natural number < 2^8
(defdata-ordered-list byte255-increasing uint8 0 255)
(defdata RequestElementType
  (record
   (ElementID . exact10)
   (Body      . byte255-increasing)))
;; ACL2s definition includes all of the constraints we had to add as
;; extra assertions in the SMTLIB model for this field!
```

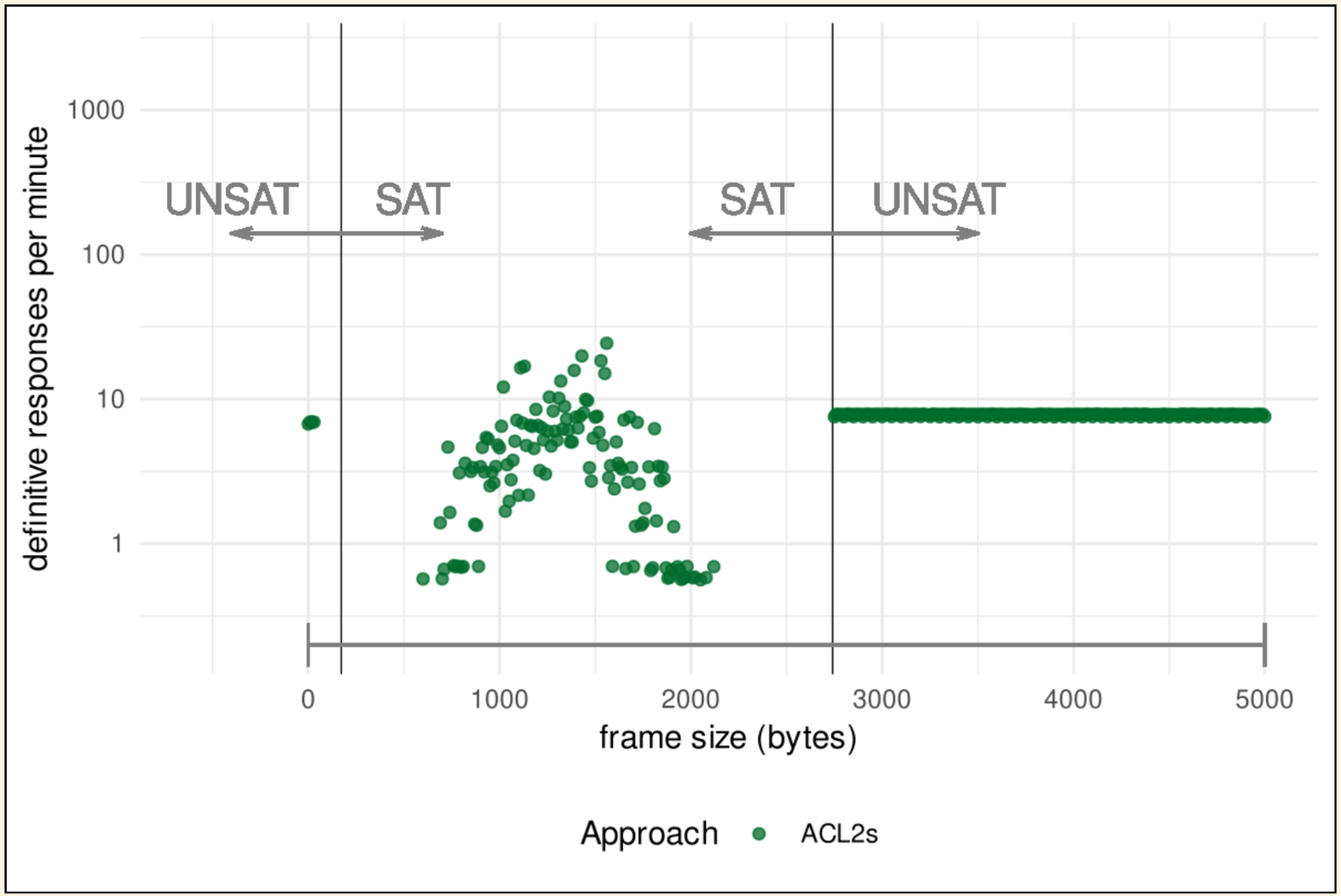
Z3 PERFORMANCE

Z3 PERFORMANCE



ACL2S PERFORMANCE

ACL2S PERFORMANCE



Enumerative Data Types

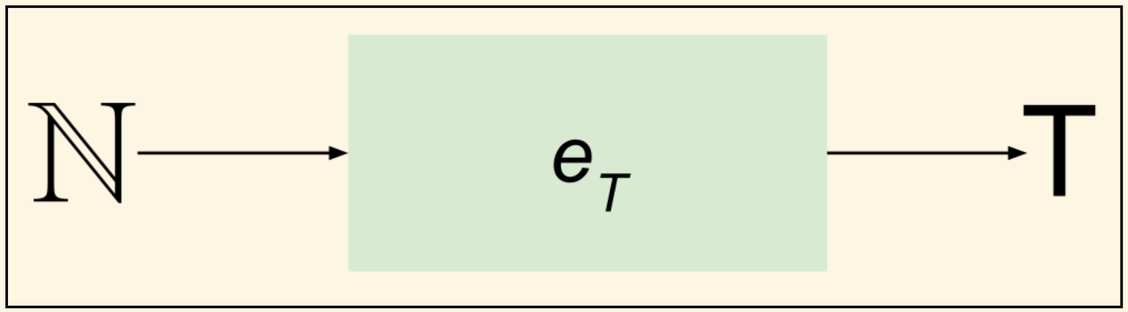
- Defdata framework automatically generates *enumerators*
- enumerator for T: $e_T : \mathbb{N} \to T$
- e_T generates elements of T entirely through computation!
- Counterexample generation framework: synergistic integration of theorem proving and testing using enumerators

Enumerators and Dependencies

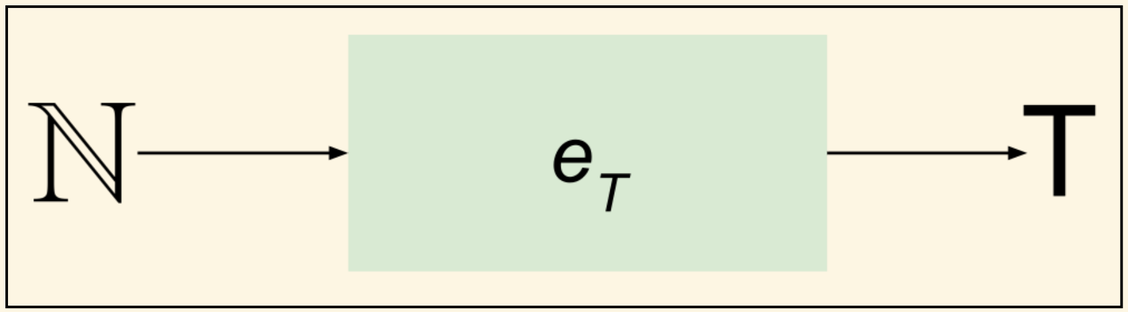
```
(acl2s-defaults :set sampling-method :uniform-random)
(defdata-list upto-10-nats nat 1 10)
(defdata foo
  (record (f1 . upto-10-nats)
          ...
          (f10 . upto-10-nats)))
(definec foo-size (x :foo) :nat
  (+ (length (foo-f1 x))
     ...
     (length (foo-f10 x))))
(property (x :foo)
  (!= (foo-size x) 100))
```

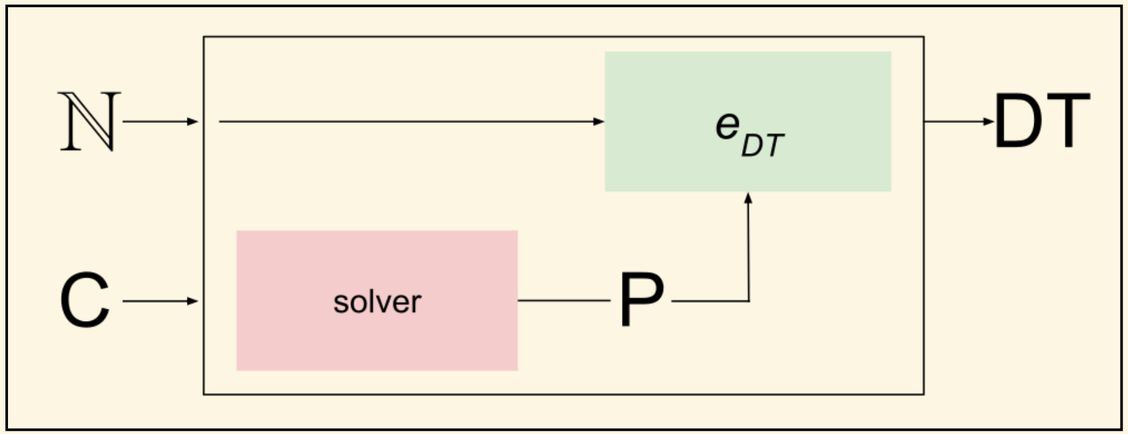
New idea: extending enumerators to dependent types

NEW IDEA: EXTENDING ENUMERATORS TO DEPENDENT TYPES

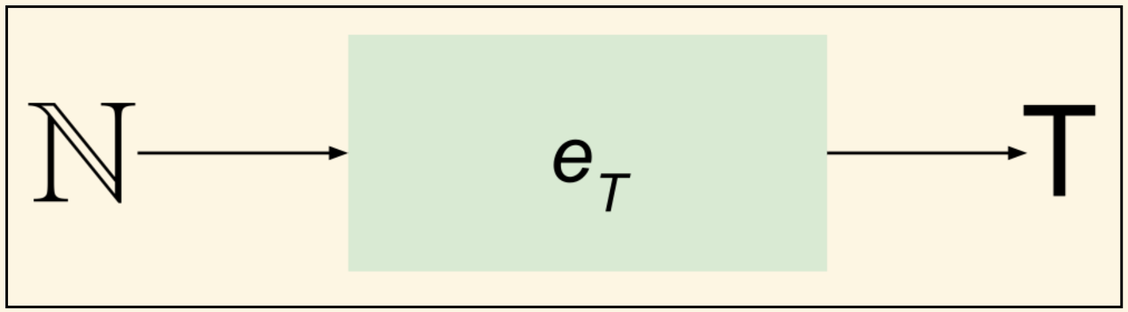


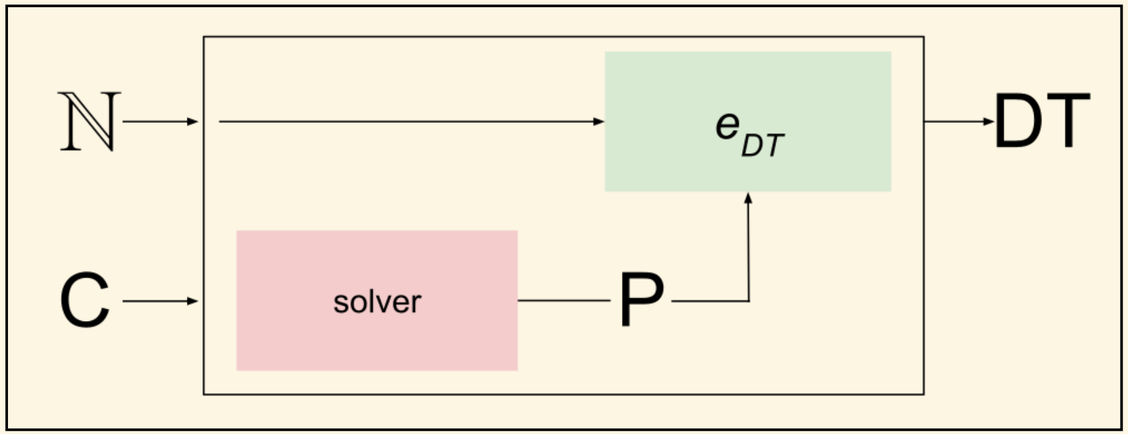
NEW IDEA: EXTENDING ENUMERATORS TO DEPENDENT TYPES





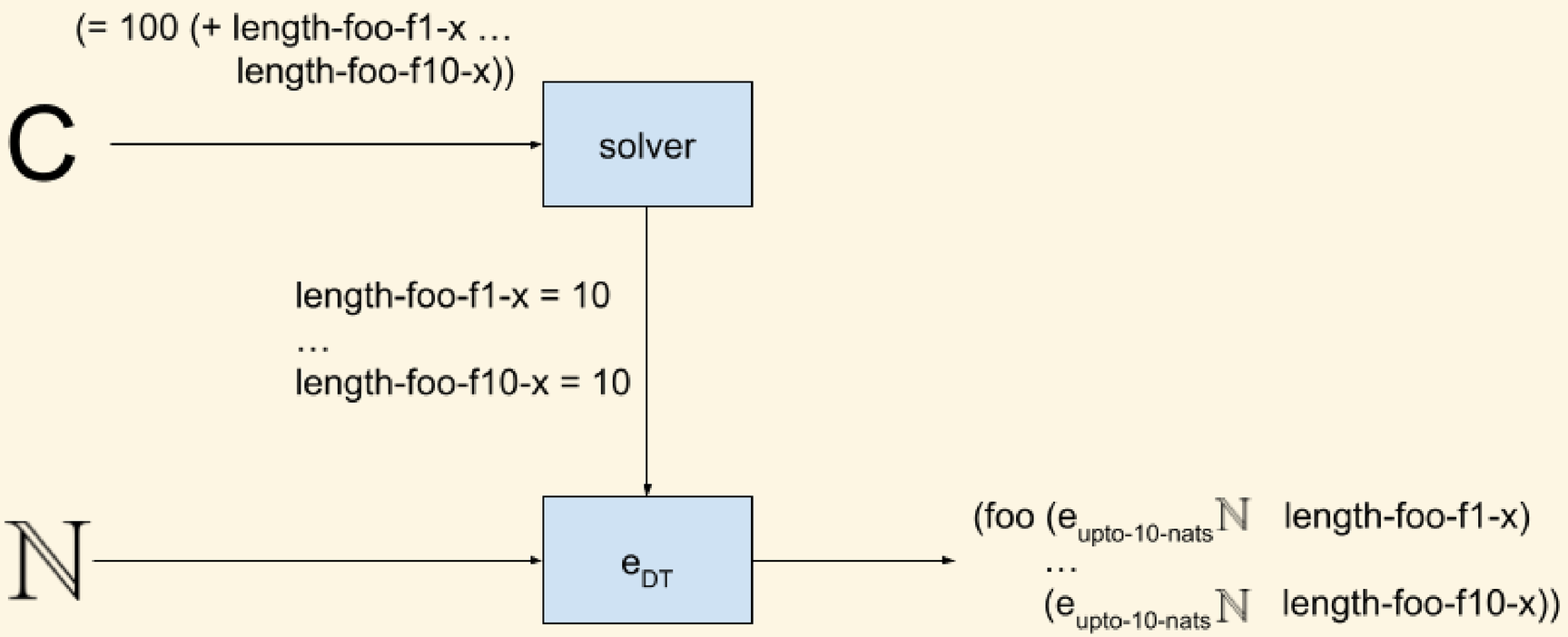
NEW IDEA: EXTENDING ENUMERATORS TO DEPENDENT TYPES



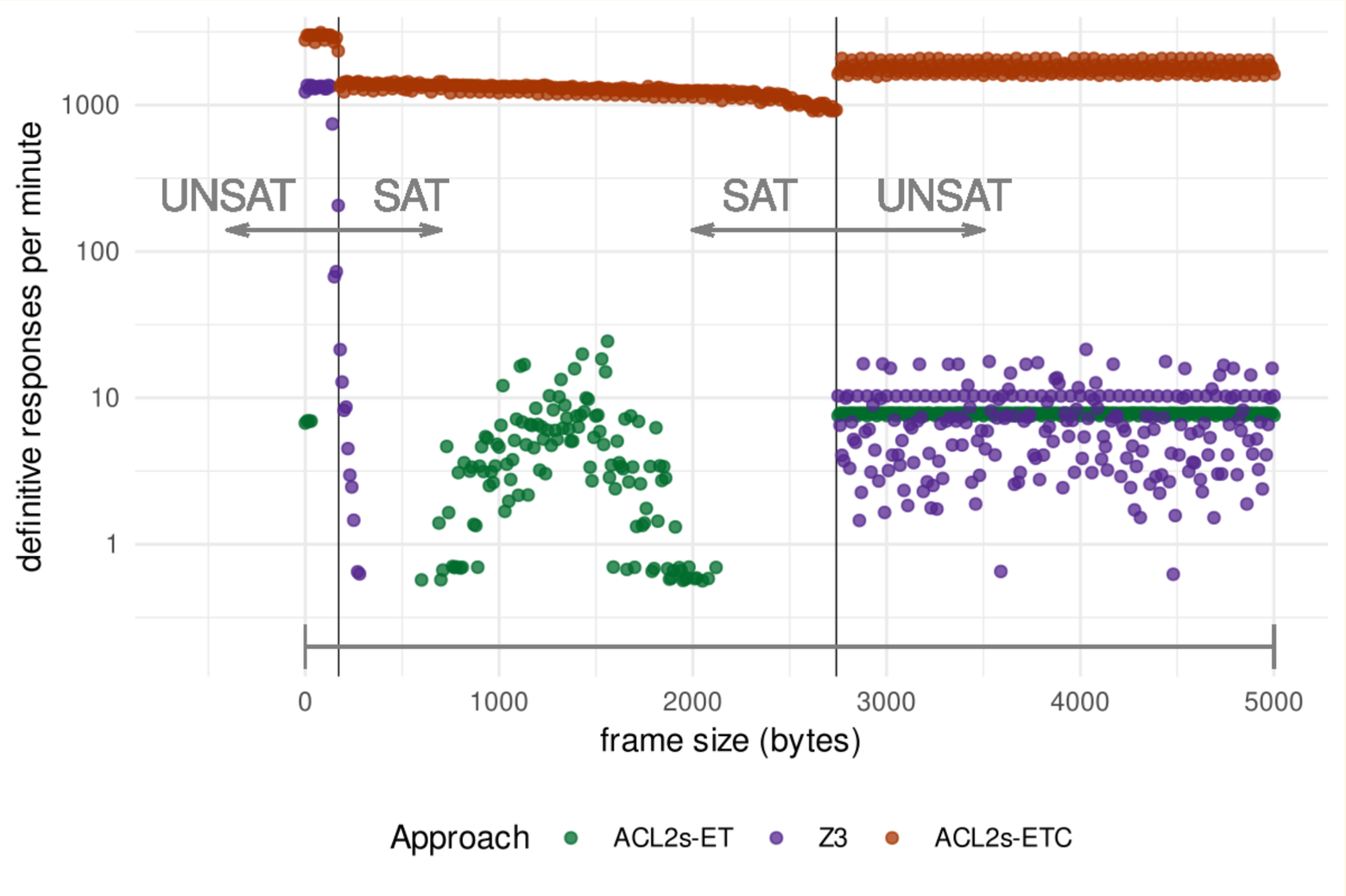


- allow users to define dependent types with *parameters*
- parameters associated with functions over values: length, etc...
- given parameter values & seed, enumerator e_{DT} can **compute** element of type DT

NEW IDEA: EXTENDING ENUMERATORS TO DEPENDENT TYPES



Enumerative Dependent Types Evaluation



FUTURE WORK

- Integrate into ACL2s
- Formalize theory for ACL2s
- Support more languages!
- Formal Methods for the masses.

Conclusion

- Enumerative dependent types are powerful!
- Minimize constraint solving, push into enumerators
- Consider using them in your tools.

QUESTIONS?