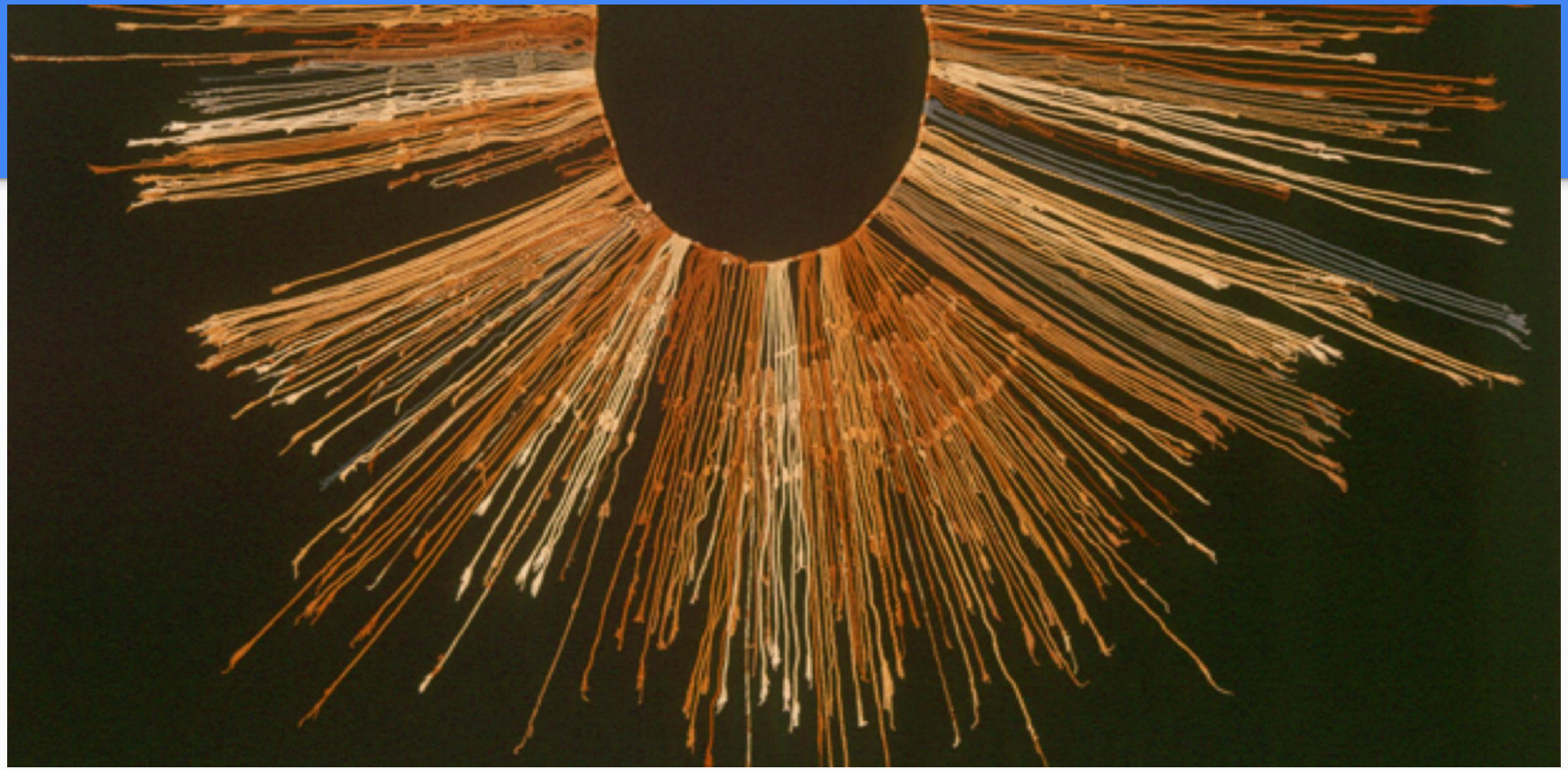
Analyzing Khipu in ACL2

Rob Sumners
Centaur Technology
rsumners@centtech.com

Brief introduction to khipu

- “Khipu” (from the Quechan for “knot”) are structured combinations of woven strings tied into knots used as a form of “writing” developed before and during the Inka empire.
- Khipu were prevalent throughout the Inka empire but almost all were destroyed by the Spanish.
- About 600 intact documented specimens in museums, estimated about 1000 specimens in total including private collections.

Pictures of khipu

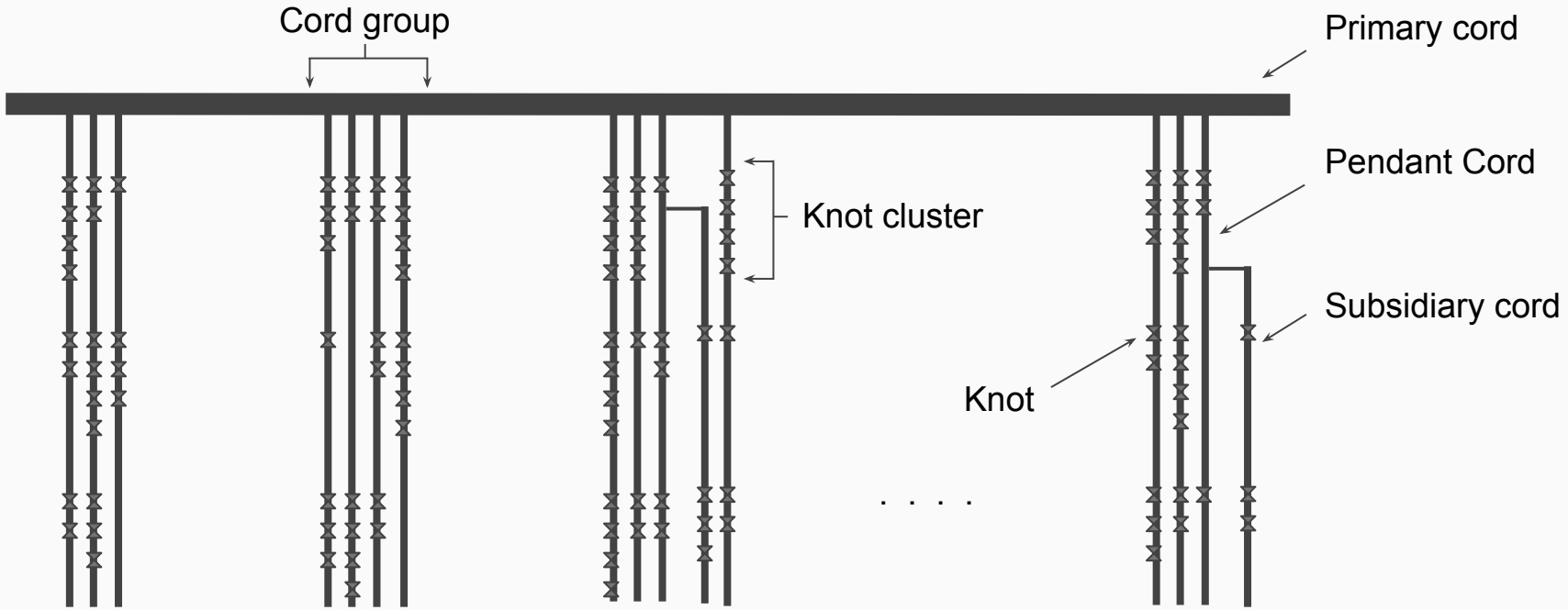


Pictures of khipu

Basic structure of khipu

- Khipu have a main or primary **cord** which forms the root of all other cords.
- Any cord can have subcords attached to it and **knots** tied into it.
- Subcord attachments may be spaced to form **cord groups** and knot placement is spaced to create **knot clusters** on a cord.
- Cords can also be differentiated by other properties such as color, material, “spin”, and attachment direction.

A more “discrete” view of a khipu



Khipu (simplified) as a recursive data type..

```
(defmacro list-of (chk name)
  `(or (equal x (quote ,name))
       (and (consp x) (let ((f (first x))) ,chk) (,name (rest x)))))

(defun knot-cluster-p (x) (list-of (knot-p f) knot-cluster-p))

(mutual-recursion
(defun cord-p (x)          (list-of (or (cord-group-p f)
                                        (knot-cluster-p f)) cord-p))
(defun cord-group-p (x)    (list-of (cord-p f) cord-group-p)))

(defun khipu-p (x)                  (cord-p x))
```

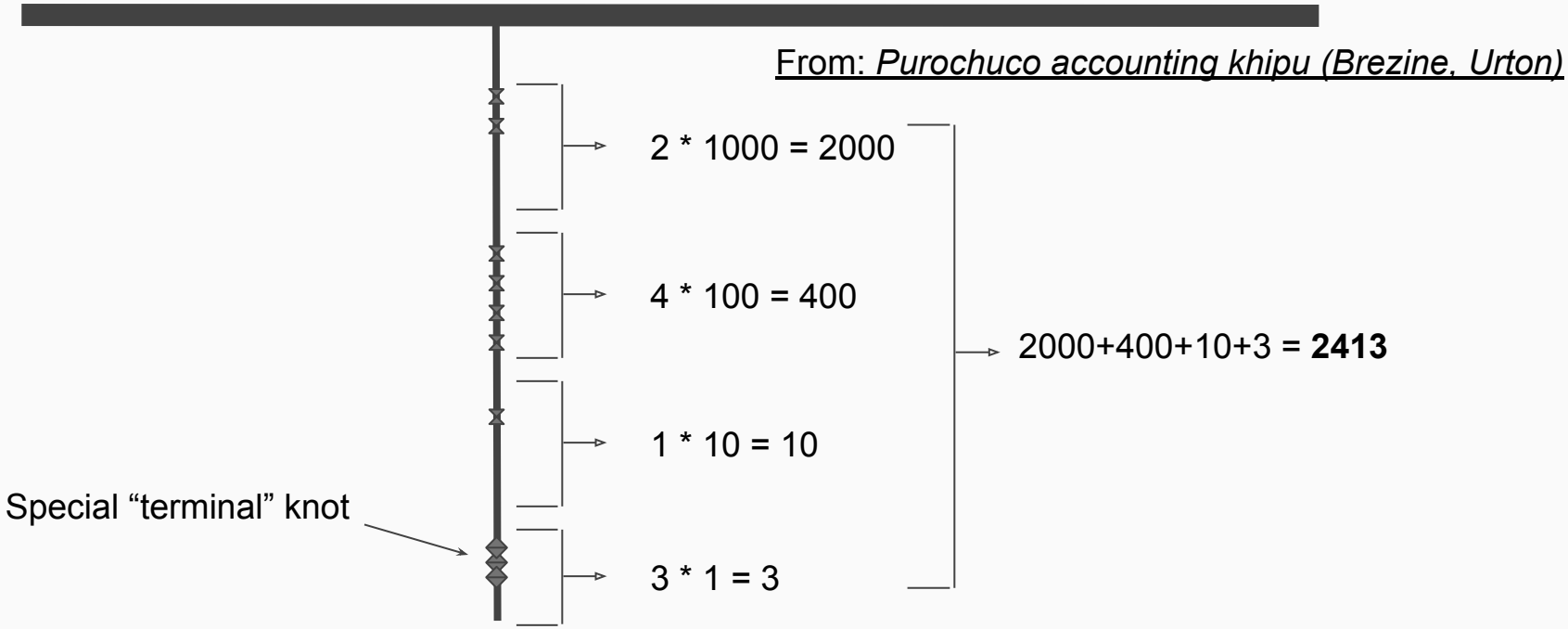
A few more notes on existing khipu..

- Numbers of pendant cords:
  - Smaller khipu have 10s of pendant cords
  - Average khipu have around 100 or so pendant cords
  - Larger khipu can have around a 1000 or more pendant cords
- Some khipu have subsidiary cords up to 10 levels deep..
  - Most khipu are only a couple levels deep
- Existing khipu have largely been found grouped in burial sites
- “Khipukamayuq” were specialists trained in producing and reading khipu

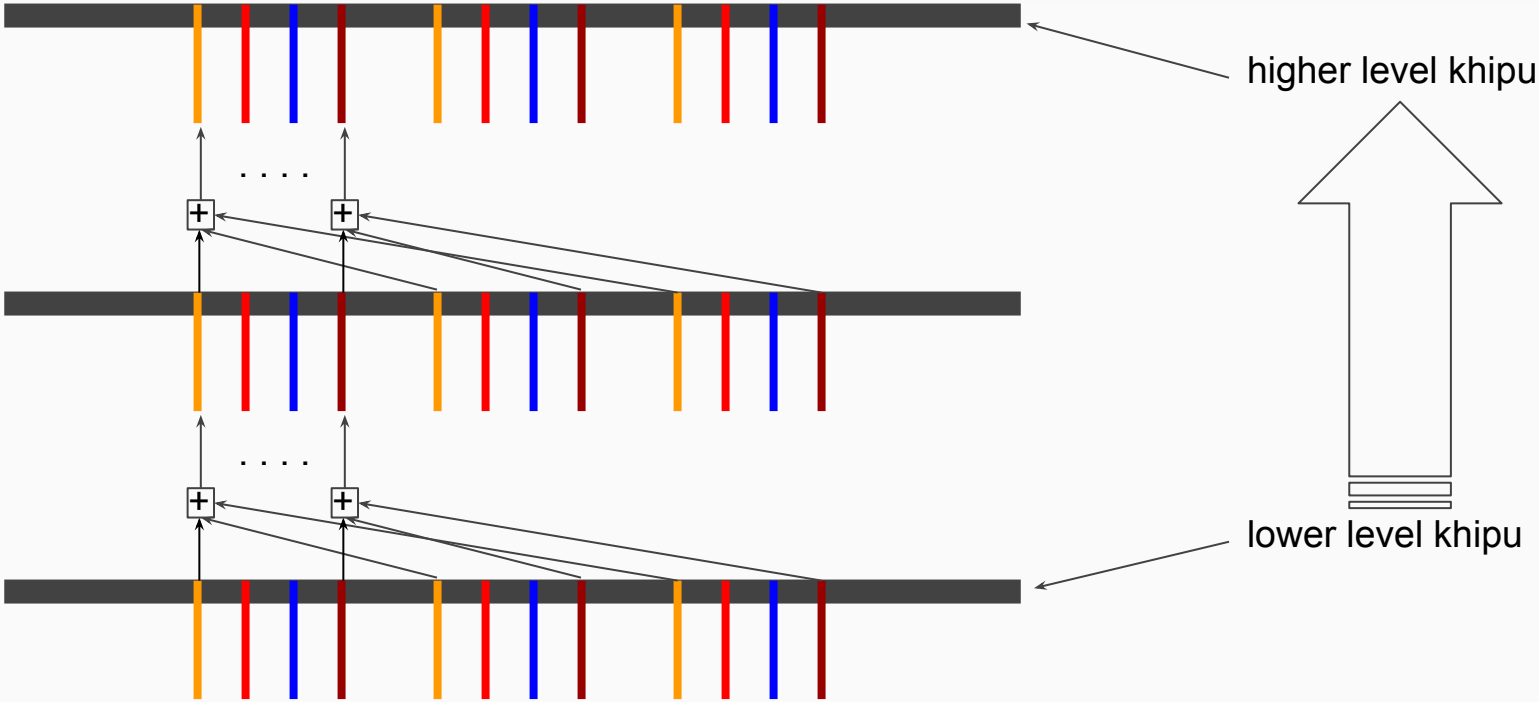
Research progress in decoding khipu

- Decoding khipu is in general unsolved but some progress has been made:
  - Decoding.knot clusters as decimal numbers (Locke)
  - Finding numerical summations across pendant cords (Ascher)
  - Correlating khipu structure to Inca bureaucratic structure (Ascher)
  - Correlating matching subcord structures to calendar seasons (Urton)
  - Matching cords as summations across khipus used for accounting (Urton, Brezine)
  - Correlating khipu data census and death records (Urton)
- Additional research into khipu which are believed to record stories, histories, and ???

Example decoding.. Numbers on a cord..



Continuing “accounting” example..
Summations across khipu..



Khipu Database Project (Urton, Brezine)

- A database collecting descriptive data on 600 (relatively) complete khipu.
    - Data for just under 50K cords and over 100K knots in these khipu.
    - About 150 fields per khipu, 100 fields per cord, 15 fields per knot.
- Database queries used by researchers to find correlations and patterns amongst khipu and to quickly test hypotheses.
- Data files can be retrieved from:

http://khipukamayuq.fas.harvard.edu

Khipu in ACL2

- Goal: to create books/definitions for processing/analyzing/theorizing about Khipu
- Current books/definitions support:
  - Read Khipu Database Project data (sql dump) files into ACL2
  - Translate khipu data into tables relating identifiers to values
  - “Compile” khipu data into stobj arrays for fast access and iteration
  - Define abstract stobj which hides array details for logical definitions
  - Define projection functions which map khipu data to tagged `khipu-p` objects
  - Checked some of the previous research results defined with ACL2 functions

Khipu in ACL2: example, cord->nums..

```
(defun cord->first-num (cord acc)
  (if (atom cord) (mv acc ())
    (let* ((fst   (first cord))
           (acc+f (+ (* acc 10) (len fst))))
      (cond
       ((terminal-knots-p fst) (mv acc+f (rest cord)))
       ((knot-cluster-p fst)   (cord->first-num (rest cord) acc+f))
       (t                      (cord->first-num (rest cord) acc))))))

(defun cord->nums (cord)
  (if (atom cord) ()
    (mv-let (first-num rest-cord) (cord->first-num cord 0)
      (cons first-num (cord->nums rest-cord)))))
```

Why Khipu in ACL2?

- First reason.. for the fun of it..
- But, in addition, ACL2 provides:
  - A clear logical picture of discrete khipu definition and properties
  - A tool for proving theorems about these definitions
  - Fast execution for testing properties on existing khipu
  - Links to SAT and SMT for checking/testing properties on a bounded set of khipu

Ongoing/future work

- Big goal: develop automated support for searching for possible relationships amongst khipu..
- Test (assumption ⇒ conclusion) as possible correlations.. where..
  - assumption and conclusion are generated logical formula of a fixed set of predicates.
  - test results and further search will be ranked based on number of existing khipu which satisfy assumption and conclusion.
- Use proven ACL2 theory on predicates to reduce generated tests and..
- Use GL/SATLINK (..maybe SMTLINK..) to further qualify/filter tests

Questions? .. and answers..

Thank you!!