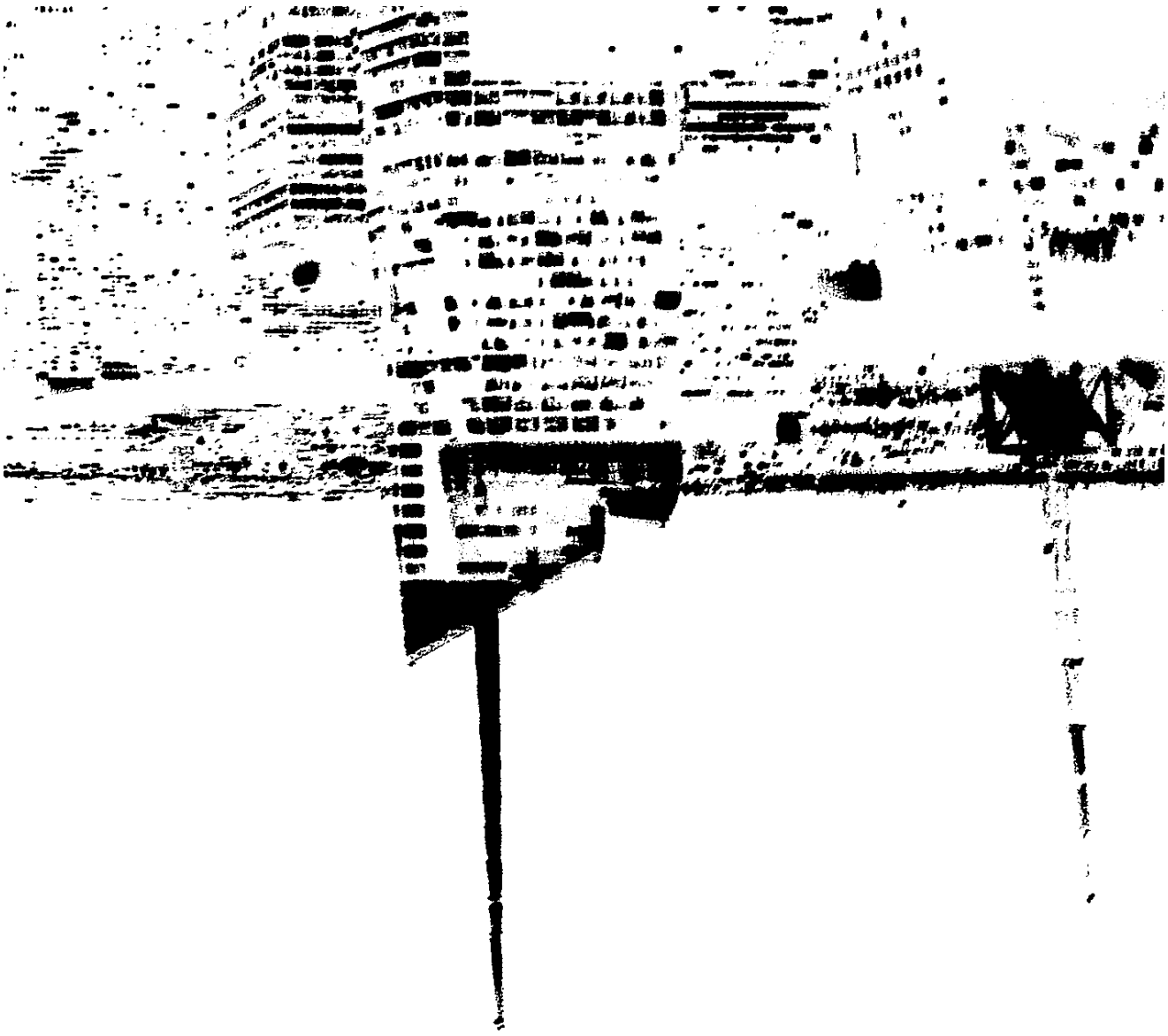

# CYBERPUNK PROPHECIES

*digital cut-up poetry*

*from 31 cyberpunk novels*

*arranged and augmented by*

**nathaniel k smith**

# cyberpunk prophecies

**cut-up poetry generated from thirty-one cyberpunk novels**

**compiled and augmented by nathaniel smith**

**using the prosaic software**

Tiny Onithopter Press
2016

First Printing: 2015, Lambaphiles Research Institute

Second Printing: April 2016, Tiny Ornithopter Press

Tiny Ornithopter Press

Portland, OR

http://graveyardtheory.net

# Preface

This is a book of cut-up poetry made from 31 Cyberpunk novels. The cut-up was performed with the help of software and augmented with original content and slight modification by a human poet.

Cyberpunk, as a subgenre, reacted to the dominant themes of Science Fiction from the 60s and 70s—American exceptionalism, white men with lasers, idyllic space travel, a march of progress realized by technological marvels—by inverting plots and putting the marginalized at the forefront of conflicts against giant corporations, corrupt governments, and out of control Artificial Intelligences.

Cyberpunk is passé for one specific reason: it became our reality. From the 1980s to the 21st century, the power of corporations and corruption of governments has grown without bound. Humans have fully entwined themselves with technology and most cannot pass a single day without looking at a computer of some kind. As a species we are looking for more novel ways to stare at computers, to put computers onto our bodies, and to put computers inside our bodies.

Reading Cyberpunk today feels simultaneously outdated and hauntingly apt. We may laugh about memory still being measured in megabytes but we shudder at Pat Cadigan's description of a limitless video network completely driven by advertising and tailored to our individualized echo chambers. These works, unlike any other genre of forward-looking fiction, have a lot to tell us about our contemporary world.

Cut-up poetry is a kind of divination. Humans seek meaning and patterns in everything and cut-up poetry is no different. By combining source texts in unexpected ways, a human observer may find meaning that both synthesizes disparate input texts and helps them understand the world around them. They may also find gibberish. It is the role of the cut-up poet to guide an essentially random process to produce works that can inspire and baffle in equal measure.

Historically, cut-up has been a manual process involving scissors and paper. However, there is now an amount of digital text so vast that no human can hope to navigate even a tiny percentage of it. We must turn to software to explore this content: only computers can dispassionately search through and analyze the amount of text we've produced. Left on their own, humans seek out the comfortable and the familiar. By working with software and large corpora of existing human works, we can produce art that challenges us in unexpected ways and helps us understand that which is locked away purely by its scale and immensity.

To cut into Cyberpunk is not an attempt to tell the future. Rather, it is an attempt to make sense of our bewildering present by cutting into the past future of prescient authors.

# Alone, the Run Could Not Be Completed

It was fucking hot in florida,
Coming in on the phone lines
With little pauses
In all the fields of dark.

The agony continued,
Taken out of porn.
The leftovers from the robot girls
Glowed pink with infrared heat.

Featureless,
It’s hard to look at anyone
Sanded free of logos.

If there were other netwalkers
Who stepped into this oven

I guess she knew why.

# Radio Signals Still Penetrated the Dead Husk

Under the spell of the vienna coven
And a focused thought from somewhere
I was waiting for a phone call.

At that moment
The opulence of the closed mall shops
Held my ability to dream.

I found my treasure
Before the machine could chime:
His memories,
Messed up with snow crash,

Blind yet seeing.

# At the Museum of Unexploded Munitions

The noise was a monster
From a lost era.

Wincing at its bite,
The guard asked
About the knot in her throat
And the socket in her neck.

Across the flat water,
Sparks of static and
Little anxieties
Reeked of lichen.

Her artificial eyes,
The unblinking eyes of the moon,
Went wild.

They watched TV
And prayed.

# Have You Heard My Startup Pitch

They walked together by habit
In only filthy coveralls
With little more sense than a rabbit
Speaking of defunct protocols.

The stuff began to twist and writhe,
Spilling over and draining out its side.
This was a most illicit enterprise
Like netsites in roma provide.

They flung it into the gulf:
A pillow upholstered in scalp.
Hands stained like the muzzle of a wolf,
They resumed kicking along the whelp.

Like businessmen draped in the pelt of an art nerd
They turned their backs on this poisoned world.

# And Through the Wires Shall Course Blood

A woman of influence,
she starts to speak.

"In the flow of the global net,
Death comes to all.

This year’s model robot
Must remove all barriers
And clean the streaks of blood.”

The wire
Tugs at her hand.

# A Fortress to House Every Crawling Thing

All sailor soldier types
They began screaming
And seemed to be headless.
They monitored the airport
And replaced it with

Office space
Any thing.

The cyborg got up
And stuck her hands
On her hips and

Radiated fracture lines
Around the chemical toilet
And through an unmarked door
At the back of the bar.

Dark silk shirts
Tan slacks,
All part of a scheme to

Blow up
This section.

The Librarian says
It was something pre captivity:
A good sign.
Where would the trigger be?

# The Cables Were Buried Beneath a Meter of Trash

She would have written programs
Behind the junk clogged storefront
Writhing like a live thing
Up geysers of fire.

The city is melting into
Cold and silent culture:
Dishes and leaking
Drug vials that move only
As in a dream.

Already it obscured her sight
Up and down the grid.
Eloquence deserted her
So she knelt
And said nothing.

It was tribal, twitchy
And would take some fancy
Shadow walking.

# Tomorrow, in a Sewer Hermitage

More of the robot baggage carts
Had become denizens of the net.

They drink from the silver palm
Assembled in orbit.

Their memories are biochemical
And their nervous systems, partial
To the shock of your generation.

Down the red wallpaper,
The jasmine smelled like musk
And junkies and prostitutes:

The comet of your disaster.

# For the First Time the Lights of the City Were Dimmed

The penile implant,
Laced through the body of the city,
Shaped the course of nations
With its wealth
Like the only people
To ever exist.

Martial law gave
A false sense of peace.
Armed fundamentalists
Told you it was dirty.

Eschewing comfort
Becoming so thin as to be intangible
And not bothering to dress
She drops silently to the ground,

Raising her captive hands
And teaching the bomb.

# The Dataset Was Adorned With Pills

The look of someone
With no dreams
Is media trance
And hurricane static.

In the new burbclaves
Scowling into the ocean wind
Longing for a bit of fun

A monster suicide rate
Put pressure on the Santa
Monica morgue.

The metacop suggests
It was a second batch of the virus
And desperately hopes it is true.

# Wires Still Trailed From the Fused Decks

I crack the seal
And half the kingdom.
Besides, the place is just
A ratmaze.

Inside, the hat man pulls off
His night glasses,
And two icons follow.

I saw her death coming
Masturbating where the best could manage.
The hacker was taken care of.
His cuffs were straightened.

I was married here,
Where the guard was back already.
She’d grown extra arms
And her dance was undulating.
Bitterly she said,

"I pay them a thing
In miniature,
A hatchet faced girl
Dancing on stage:
Our little baby
And our dirty laundry.”

It is her icon
Supposedly long retired
Activating your self destruct routine.

It’s dark for me.

# I Looked at the Text and My Eyes Were Someone Else's

She, held prisoner by his will
Took the cup held out to her.

How did you know I was online?
The skin feels no different;
I'm not even a woman,
And I don't make a lot of money.

He drew an x y graph
While the program cycled.

Their lenses went black again
Impossibly cold and distinct.
It's so hard to keep track.

Despite ambitious punks
And cut off jeans
The voice came out lost
Desperate and little.

# All That Remained Was a Flashing Prompt

The failed academic
Of clove cigarettes,
A couple of cats,
And famous landscapes
Took no visitors.

She was the light of the place:
Her oblivion chase
Crossed both beauty
And squalor.

When he ran out of ideas,
He said nothing,
And sank in the same fluid.

The blank screen
Is up to four:

Impossible to make out the end.

# Young Hands Doing Old Work

It would be nice, if
The gnawing edge of the antiseptic
Faded, this lively sense
Of the grotesque, this
Silent protest.

I like this very much:
The color of old tools
On whom I can rely
Seemed reason enough to smile.

The young apollo cast
A calm scent of coffee
From the closed door.

Fluttering one of his petals,
ICE arcs from either side,
Like coal powder streaks
Like the twenty five worst air crashes
Like the black chrome of my broken neck
Like objets d'art.

She will proceed as planned.

# Once, At a Bar in a Tokyo High Rise

Eliot and Hiro look over at Vic.
“It’s such a strange language.”
A cheap and broken bic,
Floating in drunk sewage.

It’s from Kabuki:
Flipscans in the corporate docket
Of a burned branded jet ski.
She fished a scrap out of her back pocket.

But she was lonely.
One didn’t have to wear a face mask,
To fail suddenly,
But she had decided to never ask.

"They know about your financial worth."
She shook the liquor back and forth.

# Four Haiku

"What’s that mean?"
Bundled like ganglia,
She sat beside him.

—

Austerely thin and
Riddled with cranial plugs:
He bent the console.

—

Danbala’s a program
Exhausted by the strain
Prickling down her arms

—

To make a feast of her,
Plenty of capacity:
A Maas Neotek

# Dredge Code

It’s that simple.
Your mind is made up.

On the slow boat to china,
Threads of impossible heat glimmer on
Old fashioned Windows.

She was beaming,
Here in the first world.

# It Wasn't Ours But We Lived There Anyway

At some point,
In the middle of the night,
We had our robot.

It felt more than right,
But the magic words were
Back in the loft.

With a leech DIM handy
We could deter
A world with new meaning.

# The Terminal Is Red Over Black

The Eurocops know who he is:
Their blue fatigues were spotless,
Enthusiastic.

I'd begun to choke.
But I won't be dead,
The metamartians claim.

Another dozen heartbeats,
The glowing girl said,
And then he was up.

# Seventy Dead Colonists

An icon of liberation:
Broken bottles and
Styrofoam containers.

There was a silence,
An abdication of the pioneer,
And pieces of a korean pickup.

For a while, they were right,
After splattered dripping drinks
And a jittered laser track.

The alley was empty again
And I must not port
The sick lurch of my thoughts

Or see the holo of the princess
That posed a danger
To the other power nations.

# In the Glow of the Monitors the Basement Was Bright as Day

I stared and he smiled.
From his belly extended
Shafts of light, yet he
Looked mundane, with delicate
Features and thinning hair.

He gathered up the trailing wire.
The data was neatly filed on the net
Since papyrus was perishable.

Behind the wheel,
Moral questions run
Together like gunfire.

# The Golden Haze

Light squares swarm
With golden haze
Over trampled lawns when

That fat prick
Slides up and cuts
The wire from my skull.

Old dead dishes stood
While she fumbled with the toolkit.

She looked at me fiercely
And it got my brain working again.

# I Have Bathed Your Skyscraper in Acid

Konstantin sighed.
No major debris yet.
not even rats.

A pillbox in a liquor store.
Laura half laughed and
He looked at her sardonically.

The talking head
Would be pleased, as
Everyone benefits.

It is permissible to use clean sand:
It is fine now.

# Tunnels Under Telemux

It would have been nice, but
You’ve got to keep moving.

Greta smiled, then
She moved her head.

"I got some spare virching goggles, if
You’ll follow me.”

Cobb could still see it clearly:
The name of the law on
Her other hand.

# White Screens

With one hand on the keyboard,
The screen had cleared:
Death did not come.

He sounded bored.
The nodal point was gone,
And he knew why.

The white screens behind him
Said he knew the code
Even though nothing was lit.

She would have said
To not just pound a keyboard.

Some aren’t, said Yoke,
Disturbed.

# They Traced the Wires Into the Windowless Room

She used her ISDN connection
Because it meant taking that risk.
Pausing to suck grime from her fingers
She did not understand what she herself had created.

His bones were pulled out, but
He still had his anger, fast
And ragged over the wet cobble
Of a giant pump assembly.

It was black here, and riddled with tunnels.
The coleopter whimpers in metallic pain
To the distant throbbing of music:
A wave of grief and exultation.

# A Sizable Crater

Of course, this car floats.
Gabe politely said.
While wondering which star was hers.

Lindsay was afraid, so
She read for a while.
Her hand rested on his knee.

They stank of fear:
If you desire,
A war against all.

# All of the Tripwires Make Flesh

He looked at the mushrooms,
And of the drug in the jacket
And of the derm in the cabinet.

Someone who matters,
Assertive air of ersatz authority,
They used that stuff.

Security at the morrisey.
With his other hand
Singapore is a phone call away.

# I Saw It on the TV and the TV Saw It In Me

I’d just as soon tackle Polokov
And fire an amber stream into his mouth,
Just slamming into it.

They taught that to children
The codes barely registering.
He’d get something to eat
And burst into flames
And turn towards us
Like cruel guitar feedback and
Angry bagpipes

Very likely for simstim.
I’ll settle for glamour
With a view of the imperial palace.

# The Lab Sat Underneath Thirty-two Basements

I have the standard microbes,
A mind full of little problems,
And a feeling of absurdity.
I’m sorry too.

These were japanese:
A left handed steak knife
Was an indecent suggestion.

I just made it up,
But the model runs:

Meaningful work.

# Staring Down the Barrel of a Designer Handgun

I built for investors
Things very special to me:
Drops in the consumer bucket.

I stripped off their flesh
While the officers laughed.
I tried to imagine them as a bird would.

Heedless of chemical sands
The nest inside my ear
Started to say something,
And suddenly it hit me

I won’t help you.
I am a dadaist artist.

# Appendix: Software Listing

**nyarlathotep.hy**

```
(require hy.contrib.loop)

(import [functools [partial reduce]])
(import re)
(import sys)

(import nltk)
(import [nltk.chunk :as chunk])
(import [nltk.corpus [cmudict]])

(import [util [match]])
(import [nltk-util [word->stem]])


;; # General Utility Functions
(defn invert [f] (fn [&rest args] (not (apply f args))))
(defn plus [x y] (+ x y)) ;; 2-arity for reducing
(defn comp [f g] (fn [&rest args] (f (apply g args))))

;; # NLTK Helpers
(def sd (nltk.data.load "tokenizers/punkt/english.pickle"))
(def cmudict-dict (.dict cmudict))
(def vowel-pho-re (.compile re "AA|AE|AH|AO|AW|AY|EH|EY|ER|IH|IY|OW|OY|UH|UW"))
(def vowel-re (.compile re "[aeiouAEIOU]"))
(def punct-tag-re (.compile re "^[^a-zA-Z]+$"))

(def is-vowel? (partial match vowel-re))
(def is-vowel-pho? (partial match vowel-pho-re))
(def is-punct-tag? (partial match punct-tag-re))
(def is-alpha-tag? (invert is-punct-tag?))

(defn word->phonemes [word]
  (try
   (first (get cmudict-dict (.lower word)))
   (catch [e KeyError]
     [])))  ;; Not sure what is best here...

(defn tokenize-sen [raw-text]
  (.tokenize sd raw-text))

(defn tag [sentence-str] (->> sentence-str
                              (.word-tokenize nltk)
                              (.pos-tag nltk)))

(defn word->chars [word] (list word))
(defn sentence->stems [tagged-sen]
  (->> tagged-sen
       (map (comp word->stem first))
       list))

(defn rhyme-sound [tagged-sen]
  (let [[is-punct-pair? (fn [tu] (is-alpha-tag? (second tu)))]
        [no-punct       (list (filter is-punct-pair? tagged-sen))]]
```

```
    (if (empty? no-punct)
      nil
      (let [[last-word (->> (list no-punct)
                            reversed
                            list
                            first
                            first)]
            [phonemes  (word->phonemes last-word)]]
        (.join "" (->> (reversed phonemes)
                       (take 3)
                       list
                       reversed
                       list))))))

(defn count-syllables-in-word [word]
  (let [[phonemes  (word->phonemes word)]
        [vowels (if phonemes
                  (filter is-vowel-pho? phonemes)
                  (->> (word->chars word) ;; fall back to raw vowel counting
                       (filter is-vowel?)))]]
    (len (list vowels))))

(defn count-syllables [tagged-sen]
  (->> tagged-sen
       (map first)
       (map count-syllables-in-word)
       (reduce plus)))

(defn tagged->str [tagged-sen]
  (loop [[str ""] [t-s tagged-sen]]
        (if (empty? t-s)
          str
          (let [[token-tuple (first t-s)]
                [text        (first token-tuple)]
                [tag         (second token-tuple)]
                [text        (if (is-punct-tag? tag) text (+ " " text))]]
            (recur (+ str text) (rest t-s))))))

;; # Database Interaction
(defn store! [db data] (.insert db data))

;; # Text Pre-processing
(defn multiclause? [tagged-sen]
  (some (fn [tu] (= ":" (second tu))) tagged-sen))

(defn split-multiclause [tagged-sen]
  (if (multiclause? tagged-sen)
    (let [[pred          (fn [x] (not (= ":" (second x))))]
          [first-clause  (list (take-while pred tagged-sen))]
          [second-clause (list (drop-while pred tagged-sen))]]
      [first-clause second-clause])
    [tagged-sen]))

(defn expand-multiclause [tagged-sens]
  (let [limit (len tagged-sens)]
    (loop [[t-s tagged-sens] [new []]]
          (if (empty? t-s)
            new
            (recur (rest t-s) (+ new (split-multiclause (first t-s))))))))
```

```
(defn process-sentence [tagged-sen source line-no]
  {"stems"         (sentence->stems tagged-sen)
   "source"        source
   "tagged"        tagged-sen
   "rhyme_sound"   (rhyme-sound tagged-sen)
   "phonemes"      (->> tagged-sen
                        (map first)
                        (map word->phonemes)
                        list)
   "num_syllables" (count-syllables tagged-sen)
   "line_no"       line-no
   "raw"           (tagged->str tagged-sen)})

;; # Main entry point for text processing
(defn process-txt! [raw-text source db]
  (let [[tagged-sens (->> raw-text
                          tokenize-sen
                          (map tag)
                          list
                          (expand-multiclause))]]
    (for [ix (range 0 (len tagged-sens))]
      (let [[tagged-sen (nth tagged-sens ix)]
            [to-db      (process-sentence tagged-sen source ix)]]
        (store! db to-db)))))
```

**cthulhu.hy**

```
(require hy.contrib.loop)
(loop [[x nil]] x) ; this is here to fix a threading / sys.modules issue

(import [concurrent.futures [ThreadPoolExecutor]])
(import [copy [deepcopy]])
(import [functools [partial reduce]])
(import [json [loads]])
(import sys)

(import [dogma [keyword-rule
                fuzzy-keyword-rule
                rhyme-rule
                syllable-count-rule]])
(import [util [random-nth]])

; # General Utility
(defn pluck [col-of-dicts key]
  (list (map (fn [d] (.get d key)) col-of-dicts)))
(defn merge [d0 d1]
  (let [[d0-copy (deepcopy d0)]]
    (.update d0-copy d1)
    d0-copy))

(defn merge! [d0 d1]
  (.update d0 d1)
  d0)

; # Rules and Rulesets
(defclass rule-set []
  [[rules []]
   [__init__ (fn [self rules]
               (setv (. self rules) rules)
               nil)]
   [to-query (fn [self]
               (->> (. self rules)
                    (map (fn [r] (.to-query r)))
                    (reduce merge!)))]
   [weaken! (fn [self]
              (.weaken! (random-nth (. self rules)))
              self)]])

(defn build-sound-cache [db]
  (.distinct (.find db) "rhyme_sound"))

(defn build-letters->sounds [db template &optional sound-cache]
  (let [[letters (list (set (pluck template "rhyme")))]]
    (if (empty? letters)
      {}
      (let [[sounds  (if sound-cache
                       sound-cache
                       (build-sound-cache db))]]
        (->> letters
             (map (fn [l] [l (random-nth sounds)]))
             dict)))))

(defn extract-rule [db l->s raw-pair]
  (let [[type (first raw-pair)]
        [arg  (second raw-pair)]]
```

```
    (cond
     [(= type "rhyme") (rhyme-rule (.get l->s arg))]
     [(= type "keyword") (keyword-rule arg db)]
     [(= type "syllables") (syllable-count-rule arg)]
     [(= type "fuzzy") (fuzzy-keyword-rule arg db)])))

;; need to transform dictionary -> list of rules
(defn extract-ruleset [db letters->sounds tmpl-line]
  (->> tmpl-line
       .items
       (map (partial extract-rule db letters->sounds))
       list
       rule-set))

(defn template->rulesets [db template executor &optional sound-cache]
  (let [[letters->sounds (build-letters->sounds
                          db template sound-cache)]]
  (list (.map executor (partial extract-ruleset db letters->sounds) template))))

(defn ruleset->line [db ruleset]
  (loop [[line nil] [r ruleset]]
        (let [[query (.to-query r)]
              [lines (list (.find db query))]]
          (if (empty? lines)
            (recur nil (.weaken! ruleset))
            (do
             (random-nth lines))))))

; # Main entry point
(defn poem-from-template [template db &optional sound-cache]
  (let [[executor (ThreadPoolExecutor 4)]
        [letters->sounds (build-letters->sounds db template sound-cache)]
        [poem-lines (.map executor (fn [tmpl-line]
                                     (->> tmpl-line
                                         (extract-ruleset db letters->sounds)
                                         (ruleset->line db)))
                          template)]]
    (.shutdown executor)
    (list poem-lines)))
```

**dogma.hy**

```
(import re)

(import [nltk-util [word->stem]])
(import [util [match random-nth]])

;; # General Utility
(defn smatch [pat str]
  (let [[wrapped-pat (+ ".*" pat ".*")]
        [reg-pat (.compile re wrapped-pat)]]
    (match reg-pat str)))

(defclass rule []
  [[__slots__ []]
   [strength 0]
   [weaken! (fn [self]
              (let [[str (. self strength)]]
                (if (> str 0)
                  (setv (. self strength) (dec str)))))]

   [to-query (fn [self]
               ; Typically there would be a cond over (. self
               ; strength) here, but this base class represents the
               ; trivial rule.
               {})]])

(defclass syllable-count-rule [rule]
   [[__slots__ ["syllables" "strength"]]
    [to-query (fn [self]
               (if (= 0 (. self strength))
                  (.to-query (super))
                  (let [[syllables (. self syllables)]
                        [modifier  (- syllables (. self strength))]
                        [lte       (+ syllables modifier)]
                        [gte       (- syllables modifier)]]
                    {"num_syllables" {"$lte" lte "$gte" gte}}))))]

   [__init__ (fn [self syllables]
               (setv (. self syllables) syllables)
               (setv (. self strength) syllables)
               nil)]])


(defclass keyword-rule [rule]
  [[__slots__ ["keyword" "phrase_cache" "strength"]]
   [max-strength 11]
   [where-clause-tmpl "Math.abs({} - this.line_no) <= {}"]

   [__init__ (fn [self keyword db]
               (setv (. self strength) 11)
               (setv (. self keyword) (word->stem keyword))
               (.prime-cache! self db)
               nil)]

   [prime-cache! (fn [self db]
                   (print "building phrase cache")
                   (setv (. self phrase-cache)
                         (list (.find db {"stems" (. self keyword)})))
                   (if (empty? (. self phrase-cache))
```

```
                    (setv (. self strength) 0)))]

   [to-query (fn [self]
               (if (= 0 (. self strength))
                 (.to-query (super))
                 (let [[phrase      (random-nth (. self phrase-cache))]
                       [ok-distance (- (. self max-strength)
                                       (. self strength))]
                       [line-no     (. phrase ["line_no"])]]
                   {"source" (. phrase ["source"])
                    "$where" (.format (. self where-clause-tmpl)
                                      line-no
                                      ok-distance)}))))]])

(defclass fuzzy-keyword-rule [keyword-rule]
  [[to-query (fn [self]
               (if (= 0 (. self strength))
                 (.to-query (super))
                 (let [[phrase      (random-nth (. self phrase-cache))]
                       [ok-distance (inc (- (. self max-strength)
                                            (. self strength)))]
                       [line-no     (. phrase ["line_no"])]]
                   {"source" (. phrase ["source"])
                    "line_no" {"$ne" line-no}
                    "$where" (.format (. self where-clause-tmpl)
                                      line-no
                                      ok-distance)}))))]])

(defclass rhyme-rule [rule]
  [[__slots__ ["sound" "strength"]]
   [__init__ (fn [self rhyme]
               (setv (. self strength) 3)
               (setv (. self sound) rhyme)
               nil)]
   [next-sound (fn [self]
                 (let [[str   (. self strength)]
                       [sound (. self sound)]]
                   (cond [(= 3 str) sound]
                         [(= 2 str)
                          (cond [(smatch "0" sound)
                                 (.replace sound "0" "1")]
                                [(smatch "1" sound)
                                 (.replace sound "1" "2")]
                                [(smatch "2" sound)
                                 (.replace sound "2" "0")])]
                         [(= 1 str)
                          (cond [(smatch "0" sound)
                                 (.replace sound "0" "2")]
                                [(smatch "1" sound)
                                 (.replace sound "1" "0")]
                                [(smatch "2" sound)
                                 (.replace sound "2" "1")])]))))]
   [to-query (fn [self]
               (if (= 0 (. self strength))
                 (.to-query (super))
                   {"rhyme_sound" (.next-sound self)}))]])
```

# References


Bethke, B. (1983). Cyberpunk. Amazing Science Fiction Stories, 57(4).

Bethke, B. (1997). Headcrash (Warner Books ed.). New York: Warner Books.

Cadigan, P. (2001). Dervish is digital. New York: Tor.

Cadigan, P. (1992). Fools. New York: Bantam Books.

Cadigan, P. (1991). Synners. New York: Bantam Books.

Dick, P. (1968). Do Androids Dream of Electric Sheep? New York: Doubleday.

Effinger, G. (1987). When Gravity Fails. New York: Arbor House.

Effinger, G. (1989). A Fire in the Sun. New York: Doubleday.

Effinger, G. (1991). The Exile Kiss. New York: Doubleday.

Gibson, W. (1984). Neuromancer. New York: Ace Books.t

Gibson, W. (1986). Count Zero. New York: Arbor House.

Gibson, W. (1988). Mona Lisa Overdrive. Toronto: Bantam Books.

Gibson, W. (1993). Virtual Light. New York: Bantam Books.

Gibson, W. (1996). Idoru. New York: G.P. Putnam's Sons.

Gibson, W. (1999). All Tomorrow's Parties. New York: G.P. Putnam's Sons.

Gibson, W. (2003). Pattern Recognition. New York: G.P. Putnam's Sons.

Kadrey, R. (1988). Metrophage. New York: Ace Books.

Maddox, T. (1991). Halo. New York: TOR.

Rucker, R. (1987). Software. New York: Avon.

Rucker, R. (1988). Wetware. New York: Avon Books.

Rucker, R. (1997). Freeware. New York: Avon Books.

Rucker, R. (2000). Realware. New York: EOS.

Scott, M. (1994). Trouble and Her Friends. New York: TOR.

Stephenson, N. (1992). Snow Crash. New York: Bantam Books.

Sterling, B. (1994). Heavy Weather. New York: Bantam Books.

Sterling, B. (1996). Holy Fire. New York: Bantam Books.

Sterling, B. (1988). Islands in the Net. New York: Arbor House.

Sterling, B. (1985). Schismatrix. New York: Arbor House.

Vinge, J. (1982). Psion. New York: Delacorte Press.

Williams, W. (1986). Hardwired. New York: T. Doherty Associates.

Williams, W. (1987). Voice of the Whirlwind. New York.: T. Doherty Associates.

**tyvm**

to my family for always encouraging me to be an artist above all else

and to terian for all of the <3

cut-up poetry generated from thirty-one cyberpunk novels

http://tilde.town/~vilmibm
nks@lambdaphil.es