A Computational Approach to String Figures

Yulong Liu

2023-11-29

String Figures

- Designs formed from a loop of string
- Commonly known as a children's game

String Figures

- Designs formed from a loop of string
- Commonly known as a children's game

People have been playing with the string around the world for millennia.

String Figures

- Designs formed from a loop of string
- Commonly known as a children's game

People have been playing with the string around the world for millennia.

- Entertainment during polar nights in the Arctic region

1. the native inhabitants in the arctic region play string figures for entertainment

String Figures

- Designs formed from a loop of string
- Commonly known as a children's game

People have been playing with the string around the world for millennia.

- Entertainment during polar nights in the Arctic region
- Storytelling and illustrating scenes from myths and legends

1. the indigenous people in New Zealand play string figures for storytelling and illustrating scenes from myths and legends
2. and i play string figures when overleaf takes forever to compile my slides

String Figures

- Designs formed from a loop of string
- Commonly known as a children's game

People have been playing with the string around the world for millennia.

- Entertainment during polar nights in the Arctic region
- Storytelling and illustrating scenes from myths and legends

todo insert 3 images of strings figures from the encyclopedia: the star, something i can do, something i cant do

A Computational Approach

String figures take a lot of steps to make

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

1. (take steps) for example, to make the star, we start with this initial position, and then apply two moves of picking a segment of the string

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

- Represent string figures: simple, precise

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

- Represent string figures: simple, precise
- Apply moves directly to the representations

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

- Represent string figures: simple, precise
- Apply moves directly to the representations

Motivation

A Computational Approach

1. (mot) clear way of describing how to make string figures

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

- Represent string figures: simple, precise
- Apply moves directly to the representations

Motivation

- Do string figures on paper

A Computational Approach

String figures take a lot of steps to make

- Start with an initial position
- Apply a sequence of moves
- Each move transforms a string figure to another

String figures computations

- Represent string figures: simple, precise
- Apply moves directly to the representations

Motivation

- Do string figures on paper
- Computer simulations & animations

1. (mot) teach computers how to play string figures
2. (record) store in computer, as database
3. simliar to music scores, rubiks cubes
4. one effective way of describing string figures is to draw them, as string diagrams

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

$$L1 < L2 < L3 < L4 < L5$$
$$R1 < R2 < R3 < R4 < R5$$

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

$$L1 < L2 < L3 < L4 < L5$$
$$R1 < R2 < R3 < R4 < R5$$

String segments are named by finger $F \in \{L1, \cdots, L5, R1, \cdots, R5\}$

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

$$L1 < L2 < L3 < L4 < L5$$
$$R1 < R2 < R3 < R4 < R5$$

String segments are named by finger $F \in \{L1, \cdots, L5, R1, \cdots, R5\}$

- Fn is the near string, Ff is the far string

Representation: Diagrams

Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

$$L1 < L2 < L3 < L4 < L5$$
$$R1 < R2 < R3 < R4 < R5$$

String segments are named by finger $F \in \{L1, \cdots, L5, R1, \cdots, R5\}$

- Fn is the near string, Ff is the far string
- Lp and Rp are palmar strings

1. (palmar) when we have a string between $L1$ and $L5$, we give a special name, called palmar string

Representation: Diagrams

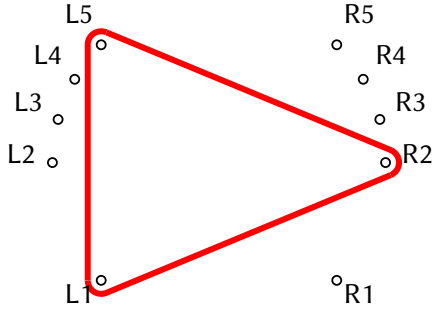
Fingers are named $L1, \cdots, L5$ and $R1, \cdots, R5$ from thumb to pinky

- Ordered from nearest to furthest

$$L1 < L2 < L3 < L4 < L5$$
$$R1 < R2 < R3 < R4 < R5$$

String segments are named by finger $F \in \{L1, \cdots, L5, R1, \cdots, R5\}$

- Fn is the near string, Ff is the far string
- Lp and Rp are palmar strings



1. the diagrams are intuitive and very visual, but they are less computer friendly
2. what do computers like? they like array of symbols

Representation: Linear Sequences

Two key components

Representation: Linear Sequences

Two key components

- Fingers that hold the string

Representation: Linear Sequences

Two key components

- Fingers that hold the string
- Crossings (with parity)

Representation: Linear Sequences

Two key components

- Fingers that hold the string
- Crossings (with parity)

Diagram $\longrightarrow$ linear sequence

Representation: Linear Sequences

Two key components

- Fingers that hold the string
- Crossings (with parity)

Diagram $\longrightarrow$ linear sequence

- Start with left nearest finger and travel clockwise

1. travel along the string clockwise

Representation: Linear Sequences

Two key components

- Fingers that hold the string
- Crossings (with parity)

Diagram $\longrightarrow$ linear sequence

- Start with left nearest finger and travel clockwise
- Visit fingers and crossings (with parity)

1. note down the fingers and crossings with parity

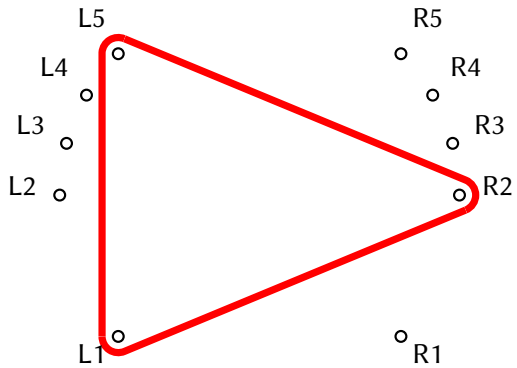
Representation: Linear Sequences

Two key components

- Fingers that hold the string
- Crossings (with parity)

Diagram $\longrightarrow$ linear sequence

- Start with left nearest finger and travel clockwise
- Visit fingers and crossings (with parity)



$\mapsto \quad L1 : L5 : R2$

Linear Sequences with Crossings

Diagram $\longrightarrow$ linear sequence

Linear Sequences with Crossings

Diagram $\longrightarrow$ linear sequence

- Name each crossing as x_i for some i

Linear Sequences with Crossings

Diagram $\longrightarrow$ linear sequence

- Name each crossing as x_i for some i
- Visit overcrossing $\implies$ write $x_i(o)$

Linear Sequences with Crossings

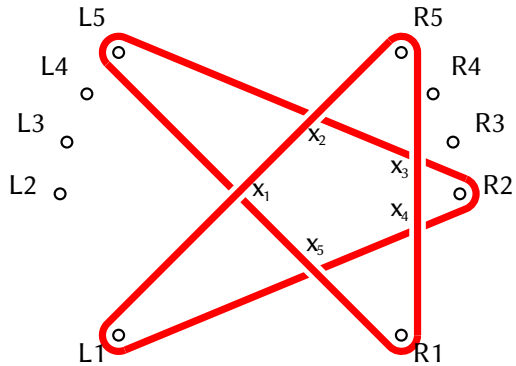
Diagram $\longrightarrow$ linear sequence

- Name each crossing as x_i for some i
- Visit overcrossing $\implies$ write $x_i(o)$
- Visit undercrossing $\implies$ write $x_i(u)$

Linear Sequences with Crossings

Diagram $\longrightarrow$ linear sequence

- Name each crossing as x_i for some i
- Visit overcrossing $\implies$ write $x_i(o)$
- Visit undercrossing $\implies$ write $x_i(u)$



$L1{:}x_1(o){:}x_2(o){:}R5{:}x_3(o){:}x_4(o){:}R1{:}x_5(o){:}x_1(u){:}L5{:}x_2(u){:}x_3(u){:}R2{:}x_4(u){:}x_5(u)$

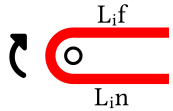
- if we have n fingers holding the string figure that has m crossings, the sequence will have $n + 2m$ symbols
- in most string figures, having a lot of crossings is what makes them beautiful, and it would be tedious to write down the entire sequence by hand
- (you see that i cant even fit the linear sequence of this simple star with normal size font)
- so this gives us more motivation to develop a systematic way of computing the movements so we can let computer do the calculuation
- the most common movements involve a finger and sometimes a near/far string segment of some other finger, but the problem arises in identifying these segments from the sequence

Identifying String Segments from Linear Sequences

Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

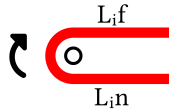
Identifying String Segments from Linear Sequences

Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

- Traverse clockwise 

Identifying String Segments from Linear Sequences

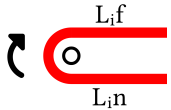
Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

- Traverse clockwise 

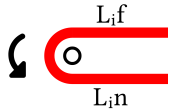
$\implies \ldots : [L_i n] L_i [L_i f] : \ldots$

Identifying String Segments from Linear Sequences

Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

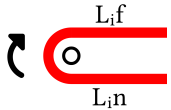
- Traverse clockwise 

$\implies \ldots : [L_i n] L_i [L_i f] : \ldots$

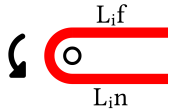
- Traverse counterclockwise 

Identifying String Segments from Linear Sequences

Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

- Traverse clockwise 

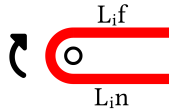
$\implies \ldots : [L_i n] L_i [L_i f] : \ldots$

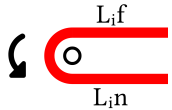
- Traverse counterclockwise 

$\implies \ldots : [L_i f] L_i [L_i n] : \ldots$

Identifying String Segments from Linear Sequences

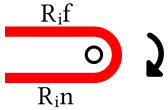
Consider a subsequence $\ldots : L_i : \ldots$ for any $L_i \in \{L1, \ldots, L5\}$

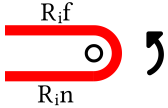
- Traverse clockwise 

$\implies \ldots : [L_i n] L_i [L_i f] : \ldots$
- Traverse counterclockwise 

$\implies \ldots : [L_i f] L_i [L_i n] : \ldots$

Similarly for finger R_i on the right hand

- 

$\implies \ldots : [R_i f] R_i [R_i n] : \ldots$
- 

$\implies \ldots : [R_i n] R_i [R_i f] : \ldots$

Identifying String Segments : Opposite Hand

Consider $\ldots : L2 : \ldots : R2 : \ldots$

Identifying String Segments : Opposite Hand

Consider $\ldots : L2 : \ldots : R2 : \ldots$

- Even number of crossings between $L2$ and $R2$ $\implies$ rotation persists

$$[L2n]\,L2\,[L2f] \;:\; [R2f]\,R2\,[R2n]$$

Identifying String Segments : Opposite Hand

Consider $\ldots : L2 : \ldots : R2 : \ldots$

- Even number of crossings between $L2$ and $R2$ $\implies$ rotation persists

$$[L2n]L2[L2f] : [R2f]R2[R2n]$$

- Odd number of crossings between $L2$ and $R2$ $\implies$ rotation reverses

$$[L2n]L2[L2f] : x_1(u) : [R2n]R2[R2f] : x_1(o)$$

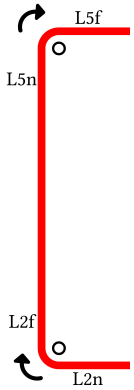
Identifying String Segments : Same Hand

Consider $\ldots : L2 : \ldots : L5 : \ldots$

Identifying String Segments : Same Hand

Consider $\ldots : L2 : \ldots : L5 : \ldots$

Even $\Longrightarrow$ rotation persists

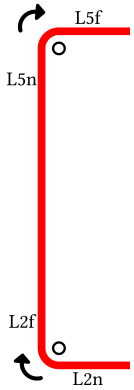


$\ldots : [L2n]\, L2\, [L2f] : [L2n]\, L2\, [L2f] : \ldots$

Identifying String Segments : Same Hand

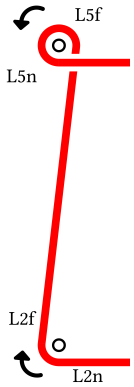
Consider $\ldots : L2 : \ldots : L5 : \ldots$

Even $\Longrightarrow$ rotation persists



$\ldots : [L2n]L2[L2f] : [L2n]L2[L2f] : \ldots$

Odd $\Longrightarrow$ rotation reverses



$\ldots : [L2n]L2[L2f] : x_1(u) : [L5f]L5[L5n] : x_1(o) : \ldots$

Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$L1\colon x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

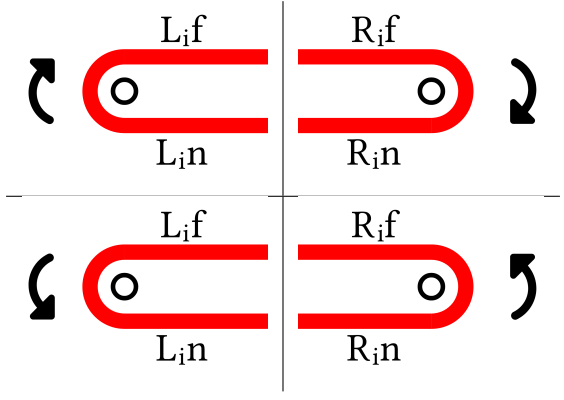
$$\widetilde{L1} : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]\ \widetilde{L1}\ [f] : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

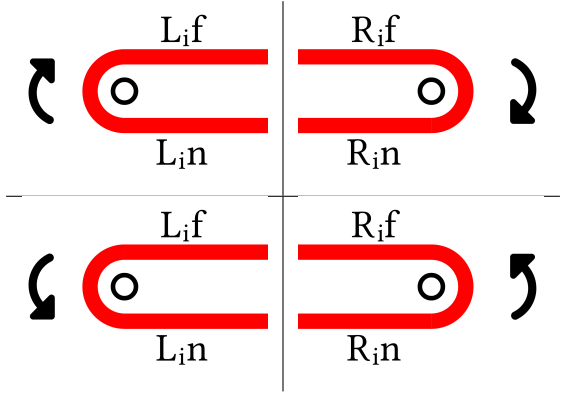


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]\ \widetilde{L1}\ [f] : x_1(o) : \overset{\curvearrowleft}{R5} : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

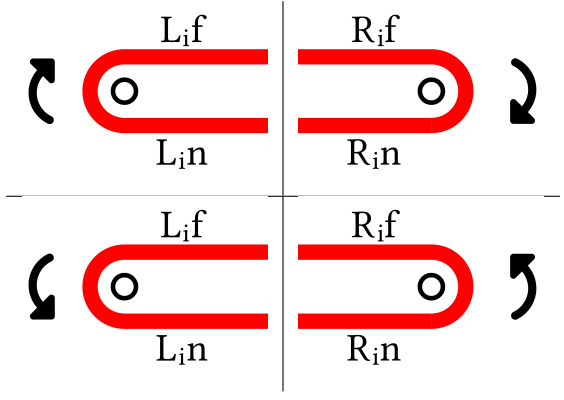


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]\ \widetilde{L1}\ [f] : x_1(o) : [n]\ \overset{\curvearrowleft}{R5}\ [f] : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

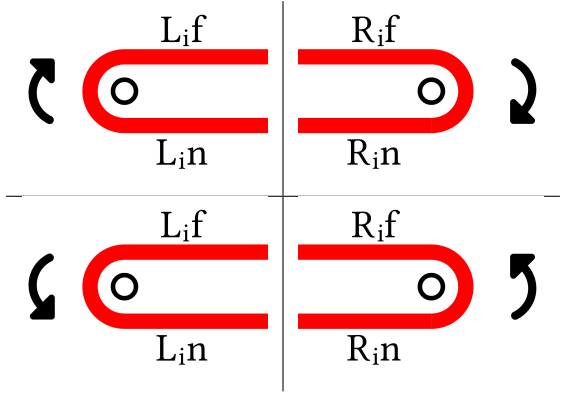


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]L1[f] : x_1(o) : [n]\ \overset{\curvearrowleft}{R5}\ [f] : x_2(o) : \overset{\curvearrowright}{L5} : x_2(u) : x_1(u) : R2$$

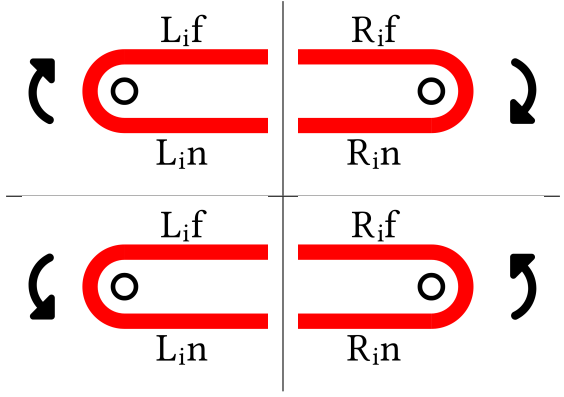


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]L1[f] : x_1(o) : [n]\ \overset{\curvearrowleft}{R5}\ [f] : x_2(o) : [n]\ \overset{\curvearrowright}{L5}\ [f] : x_2(u) : x_1(u) : R2$$

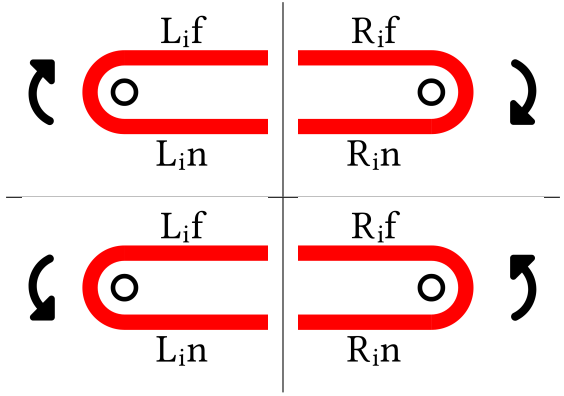


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]L1[f] : x_1(o) : [n]R5[f] : x_2(o) : [n]\ \overset{\curvearrowright}{L5}\ [f] : x_2(u) : x_1(u) : \overset{\curvearrowright}{R2}$$

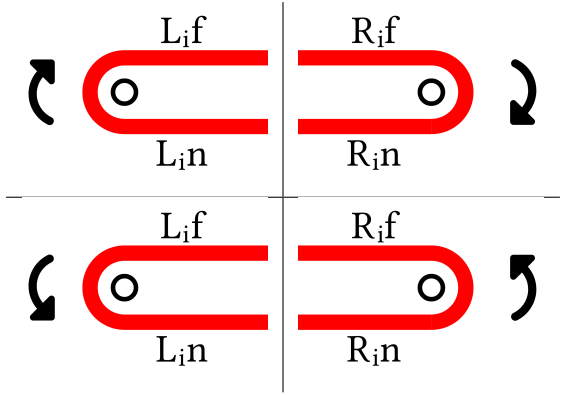


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]L1[f] : x_1(o) : [n]R5[f] : x_2(o) : [n]\ \widetilde{L5}\ [f] : x_2(u) : x_1(u) : [f]\ \widetilde{R2}\ [n]$$

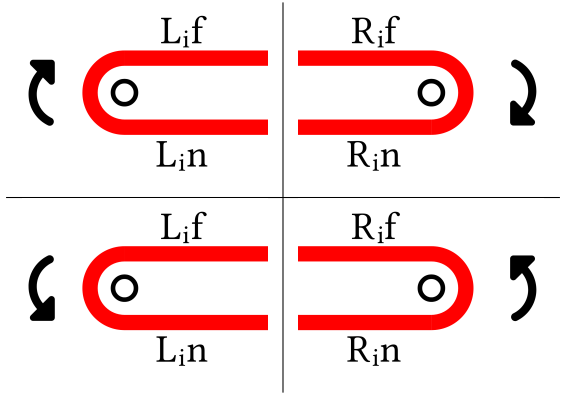


Identifying String Segments : Example

$$L1 : x_1(o) : R5 : x_2(o) : L5 : x_2(u) : x_1(u) : R2$$

By convention, the first finger in the linear sequence is clockwise

$$[n]L1[f] : x_1(o) : [n]R5[f] : x_2(o) : [n]L5[f] : x_2(u) : x_1(u) : [f]R2[n]$$



- now that we can recover the string segments from the linear sequence
- when we want to pick a specific string segment, we know where it is in the linear sequence
- "use a finger to pick some segment" is essentially insert the finger to where the segment is in the linear sequence along with some crossings if needed

Moves: Twist

Moves: Twist

Two variations: twist towards and twist away

Moves: Twist

Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$

Moves: Twist

Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$
- Twist the loop on finger F *away* from player: $> F$

Moves: Twist

Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$
- Twist the loop on finger F *away* from player: $> F$

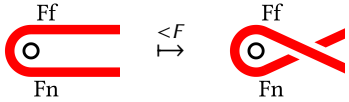
Consider . . . : $[Fn]F[Ff]$: . . .

Moves: Twist

Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$
- Twist the loop on finger F *away* from player: $> F$

Consider $\ldots : [Fn]F[Ff] : \ldots$

- 

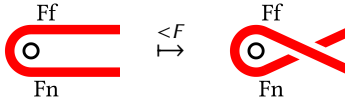
$$\ldots : [Fn]F[Ff] : \ldots \quad \overset{<F}{\mapsto} \quad \ldots : x(u) : F : x(o) : \ldots$$

Moves: Twist

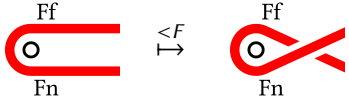
Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$
- Twist the loop on finger F *away* from player: $> F$

Consider $\ldots : [Fn]F[Ff] : \ldots$

- 

  $$\ldots : [Fn]F[Ff] : \ldots \overset{<F}{\mapsto} \ldots : x(u) : F : x(o) : \ldots$$

- 

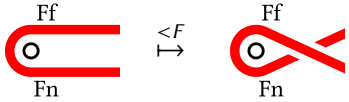
  $$\ldots : [Fn]F[Ff] : \ldots \overset{>F}{\mapsto} \ldots : x(o) : F : x(u) : \ldots$$

Moves: Twist

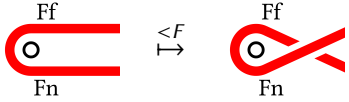
Two variations: twist towards and twist away

- Twist the loop on finger F *towards* player: $< F$
- Twist the loop on finger F *away* from player: $> F$

Consider $\ldots : [Fn]F[Ff] : \ldots$

- 

  $$\ldots : [Fn]F[Ff] : \ldots \overset{<F}{\mapsto} \ldots : x(u) : F : x(o) : \ldots$$

- 

  $$\ldots : [Fn]F[Ff] : \ldots \overset{>F}{\mapsto} \ldots : x(o) : F : x(u) : \ldots$$

$\ldots : [Ff]F[Fn] : \ldots \implies$ crossing parities are reversed

- works on right hand too
- Storer p382

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments
- F picks s from *above/below*

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments
- F picks s from *above/below*

Examples

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments
- F picks s from *above/below*

Examples

- "$R5$ passes *over* all intermediate segments and picks Lp from *above*" is denoted as $\overleftarrow{R5}(\overline{Lp})$

Moves: Pick

1. the first 2 moves makes the star

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments
- F picks s from *above/below*

Examples

- "$R5$ passes *over* all intermediate segments and picks Lp from *above*" is denoted as $\overleftarrow{R5}(\overline{Lp})$
- "$R1$ passes *over* all intermediate segments and picks $R5n$ from *below*" is denoted as $\overrightarrow{R1}(\underline{R5n})$

Moves: Pick

Finger F picks a string segment s

- Written as $F(s)$

Two types of variations:

- F passes *over/under* all intermediate segments
- F picks s from *above/below*

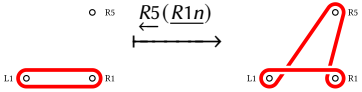
Examples

- "$R5$ passes *over* all intermediate segments and picks Lp from *above*" is denoted as $\overleftarrow{R5}(\overline{Lp})$
- "$R1$ passes *over* all intermediate segments and picks $R5n$ from *below*" is denoted as $\overrightarrow{R1}(\underline{R5n})$
- "$R4$ passes *below* all intermediate segments and picks $L1n$ from *below*" is denoted as $\overleftarrow{R4}(\underline{L1n})$

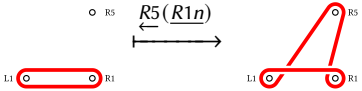
1. pick moves can be really hard to perform in reality, e.g. apply 3rd one after 1 and 2

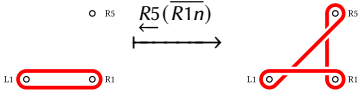
Pick: Examples

Pick: Examples

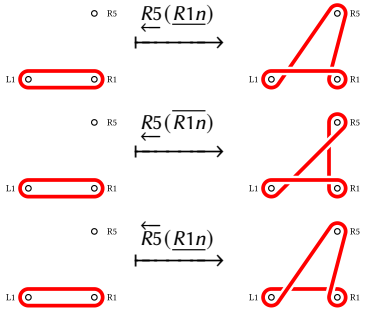


Pick: Examples

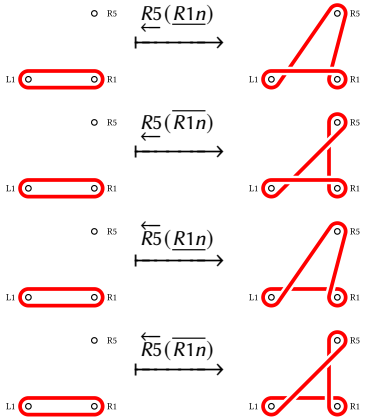




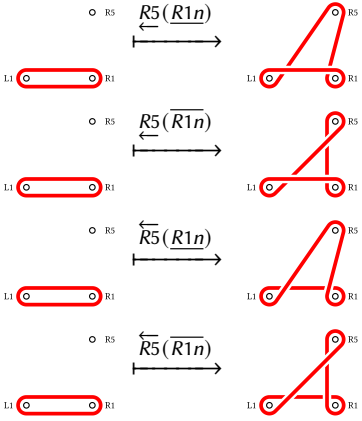
Pick: Examples



Pick: Examples

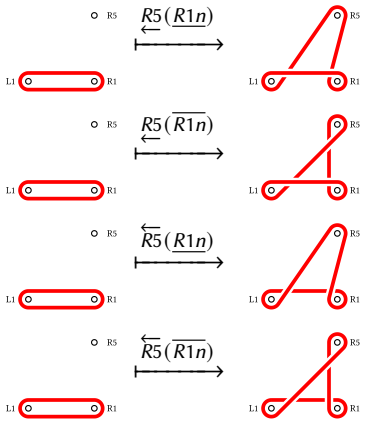


Pick: Examples



Observations

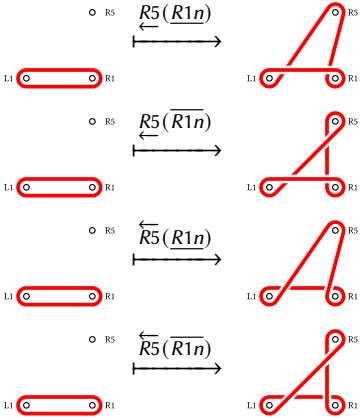
Pick: Examples



Observations

- A pair of crossings for each intermediate string

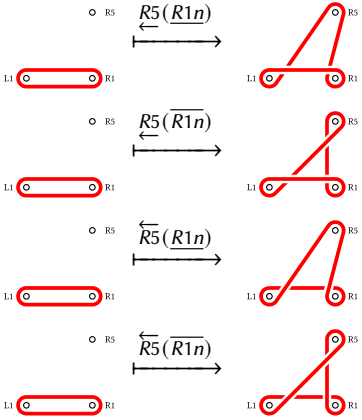
Pick: Examples



Observations

- A pair of crossings for each intermediate string
- $F(\overline{s})$ and $F(\underline{s})$ differ by a twist

Pick: Examples



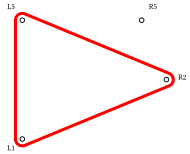
Observations

- A pair of crossings for each intermediate string
- $F(\overline{s})$ and $F(\underline{s})$ differ by a twist
- $\overleftarrow{F}(s)$ and $\underleftarrow{F}(s)$ differ by crossing parity

- when F moves towards, the new twist is a twist towards
- when F moves away, the new twist is away, e.g. $L5 : R5$ and pick with $R1$

Pick: Construction

General steps for $F(s)$

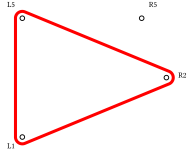


1. note that we can check if a move is valid since we know how to identify all the segments in a seq

Pick: Construction

General steps for $F(s)$

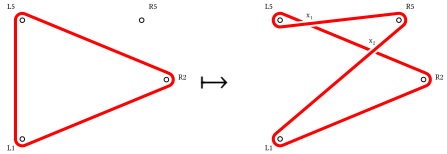
- Identify intermediate segments ($Fn < F < Ff$)



Pick: Construction

General steps for $F(s)$

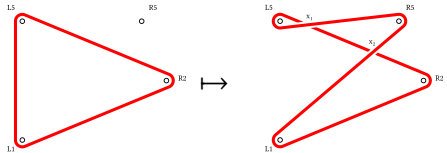
- Identify intermediate segments ($Fn < F < Ff$)
- Insert a pair of crossings for each intermediate segment



Pick: Construction

General steps for $F(s)$

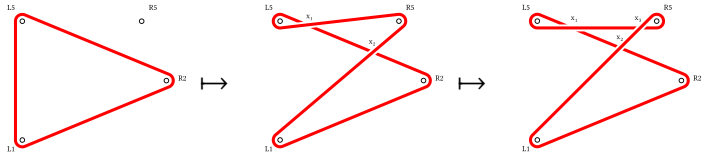
- Identify intermediate segments ($Fn < F < Ff$)
- Insert a pair of crossings for each intermediate segment
- Insert F at s with crossings (aka Spike)



Pick: Construction

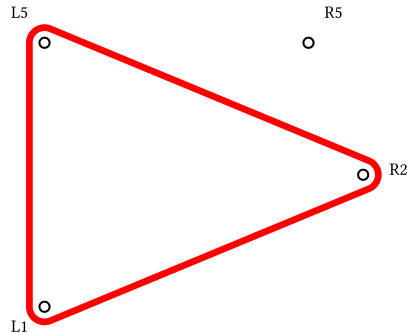
General steps for $F(s)$

- Identify intermediate segments ($Fn < F < Ff$)
- Insert a pair of crossings for each intermediate segment
- Insert F at s with crossings (aka Spike)
- Add twist if pick from above



Pick: Construction Example

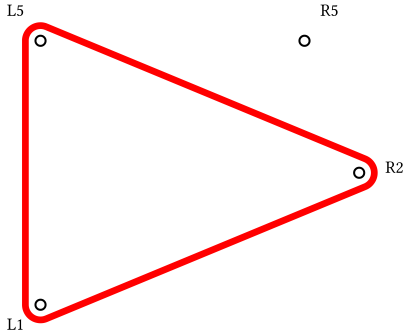
$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$



Pick: Construction Example

$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$

- $[L1n]L1[Lp]L5[L5f] : [R2f]R2[R2n]$

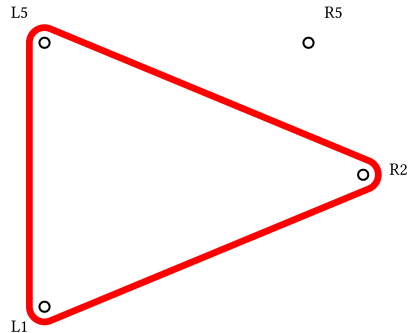


Pick: Construction Example

$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$

- $[L1n]L1[Lp]L5[L5f] : [R2f]R2[R2n]$
- Only the segment $[L5f] : [R2f]$ is intermediate

$$Lp = L5n < L5f = R2f < R2 < R5$$



Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5[L5f] : [R2f]R2[R2n]$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5[L5f] : x_1(u) : x_2(u) : [R2f]R2[R2n]$$

1. under because our finger is going above

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$L1[Lp]L5 : R2$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$\overset{\curvearrowright}{L1}: x_2(o) : \overset{\curvearrowleft}{R5}: x_1(o) : \overset{\curvearrowright}{L5}: x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Make twist on $R5$ (towards)

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Make twist on $R5$ (towards)

$$L1 : x_2(o) : x_3(o) : R5 : x_3(u) : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$L1 : L5 : R2 \xmapsto{\overleftarrow{R5}(\overline{Lp})} ???$

Found $[L5f] : [R2f]$ as an intermediate segment

- Insert crossings x_1 and x_2

$$L1 : L5 : x_1(u) : x_2(u) : R2$$

Make spike for $R5$ and insert at Lp

- Spike: $x_2(o) : R5 : x_1(o)$

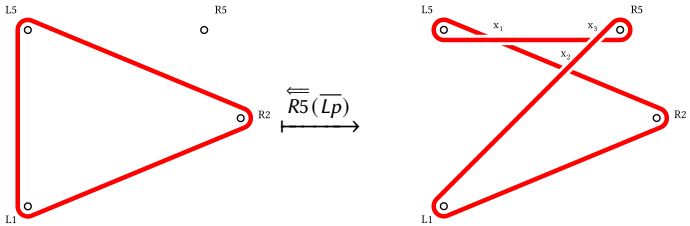
$$L1 : x_2(o) : R5 : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Make twist on $R5$ (towards)

$$L1 : x_2(o) : x_3(o) : R5 : x_3(u) : x_1(o) : L5 : x_1(u) : x_2(u) : R2$$

Pick: Construction Example

$$L1{:}L5{:}R2 \xmapsto{\overleftarrow{R5}\,(\overline{Lp})} L1{:}x_2(o){:}x_3(o){:}R5{:}x_3(u){:}x_1(o){:}L5{:}x_1(u){:}x_2(u){:}R2$$



Extension Cancellation