CS 383

Lecture 16 – Turing machine variants

Stephen Checkoway

Spring 2024

Basic Turing machine

The basic formulation of a TM

- must move its head either left or right after each step;
- has a single tape; and
- is deterministic: at each step, the TM has no choice in what action to take

A natural question: What happens if we relax these restrictions? Is the resulting machine more powerful? (I.e., can it recognize more languages?)

Turing machine with stay put

A TM with stay put is a TM that is allowed to keep the tape head in place in addition to moving left or right

The transition function changes to

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{S}, \mathrm{R}\}$$

What class of languages do TMs with stay put recognize?

How do we even address questions like this?

This is really a question about equalities of sets (usually we say classes).

$$\text{FINITE} = \{L \mid \text{the language } L \text{ has a finite number of strings}\}$$
$$\text{REGULAR} = \{L \mid \text{the language } L \text{ is recognized by some DFA}\}$$
$$\text{CONTEXT-FREE} = \{L \mid \text{the language } L \text{ is generated by some CFG}\}$$
$$\text{DECIDABLE} = \{L \mid \text{the language } L \text{ is decided by some TM}\}$$
$$\text{TURING-RECOGNIZABLE} = \{L \mid \text{the language } L \text{ is recognized by some TM}\}$$

Note that these are sets of sets (sets of languages)

How do we even address questions like this?

This is really a question about equalities of sets (usually we say classes).

$$\textsc{Finite} = \{L \mid \text{the language } L \text{ has a finite number of strings}\}$$
$$\textsc{Regular} = \{L \mid \text{the language } L \text{ is recognized by some DFA}\}$$
$$\textsc{Context-free} = \{L \mid \text{the language } L \text{ is generated by some CFG}\}$$
$$\textsc{Decidable} = \{L \mid \text{the language } L \text{ is decided by some TM}\}$$
$$\textsc{Turing-recognizable} = \{L \mid \text{the language } L \text{ is recognized by some TM}\}$$

Note that these are sets of sets (sets of languages)

We've talked about these before without really writing down specifically what they mean:

- "The class of regular languages is closed under union" means
  "If $A, B \in \textsc{Regular}$, then $A \cup B \in \textsc{Regular}$
- "The class of context-free languages is closed under Kleene star" means
  "If $A \in \textsc{Context-free}$, then $A^* \in \textsc{Context-free}$

Equality of sets

If we want to show that two sets X and Y are equal, we show two things

- $X \subseteq Y$ and
- $Y \subseteq X$

Equality of sets

If we want to show that two sets X and Y are equal, we show two things

- $X \subseteq Y$ and
- $Y \subseteq X$

We've done this over and over when making claims about languages:

Equality of sets

If we want to show that two sets X and Y are equal, we show two things

- $X \subseteq Y$ and
- $Y \subseteq X$

We've done this over and over when making claims about languages:

- "If $x \in A$, then ... thus M accepts x" is showing $A \subseteq L(M)$

Equality of sets

If we want to show that two sets X and Y are equal, we show two things

- $X \subseteq Y$ and
- $Y \subseteq X$

We've done this over and over when making claims about languages:

- "If $x \in A$, then ... thus M accepts x" is showing $A \subseteq L(M)$
- "If M accepts x, then ... thus $x \in A$" is showing $L(M) \subseteq A$

Equality of sets

If we want to show that two sets X and Y are equal, we show two things

- $X \subseteq Y$ and
- $Y \subseteq X$

We've done this over and over when making claims about languages:

- "If $x \in A$, then ... thus M accepts x" is showing $A \subseteq L(M)$
- "If M accepts x, then ... thus $x \in A$" is showing $L(M) \subseteq A$

Sometimes we prove the contrapositive of those statements:

- "If M rejects x, then ... thus $x \notin A$"
- "If $x \notin A$, then ... thus M rejects x"

So what about TMs with stay put?

After a little thinking, we might claim "the class of languages recognized by TMs with stay put is the class of Turing-recognizable languages"

So what about TMs with stay put?

After a little thinking, we might claim "the class of languages recognized by TMs with stay put is the class of Turing-recognizable languages"

To prove this, we need to show two things:

1. Every Turing-recognizable (RE) language can be recognized by a TM with stay put
2. Every language recognized by a TM with stay put can be recognized by a TM

Simulation

To do this, we need to start with a machine of one type and use it to simulate a machine of the other type

Sometimes this is easy, sometimes it is quite involved

Simulating a TM using a TM with stay put

Theorem

If A is Turing-recognizable (RE), then A is recognized by a TM with stay put

How do we prove this?

Simulating a TM using a TM with stay put

Theorem

If A is Turing-recognizable (RE), then A is recognized by a TM with stay put

How do we prove this?

Proof.

If A is RE, then there is some (basic) TM M that recognizes it.

A basic TM *is* a TM with stay put that just happens to not ever use the stay put feature.

Thus, M is a TM with stay put that recognizes A. □

Simulating a TM with stay put using a TM

Theorem

If A is a language recognized by a TM with stay put, then A is recognized by a (basic) TM.

How do we prove this?

Simulating a TM with stay put using a TM

Theorem

If A is a language recognized by a TM with stay put, then A is recognized by a (basic) TM.

How do we prove this? Proof. Let M be a TM with stay put that recognizes A. We're going to build a basic TM M' that recognizes A.

At a high level, every time M stays put, M' will move right and then back left.

Simulating a TM with stay put using a TM

Theorem

If A is a language recognized by a TM with stay put, then A is recognized by a (basic) TM.

How do we prove this? Proof. Let M be a TM with stay put that recognizes A. We're going to build a basic TM M' that recognizes A.

At a high level, every time M stays put, M' will move right and then back left.

We can be explicit about our construction in terms of the 7-tuples for M and M' but let's be *slightly* higher level than that:

For each state $q \in Q$, add a state q' to M' along with transitions $\delta'(q', t) = (q, t, \mathrm{L})$ for each $t \in \Gamma$.

For each transition $\delta(q, t) = (r, s, \mathrm{S})$ in M, replace that with $\delta'(q, t) = (r', s, \mathrm{R})$.

Proof continued

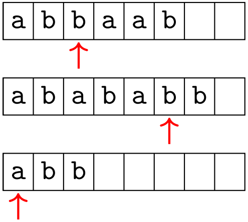
When M and M' run on some input w, every time M would transition to state q and stay put, M' will transition to q' and move right, and then transition to q and move left.

M and M' have exactly the same behavior on every string: either both accept, both reject, or both loop so $L(M') = L(M) = A$. □

Putting the two proofs together, we have shown that the class of languages recognized by a TM with stay put is the class of Turing-recognizable languages

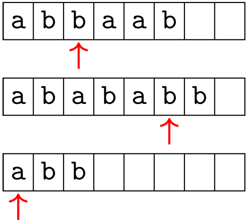
Multitape Turing machine

A k-tape TM (written k-TM) is a TM with k tapes and k independent tape heads



Multitape Turing machine

A k-tape TM (written k-TM) is a TM with k tapes and k independent tape heads



Mathematically, it's 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$

The transition function has to change to accommodate the multiple tapes

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{\mathrm{L}, \mathrm{S}, \mathrm{R}\}^k$$

Multitape Turing machine computation

When a k-TM starts, its input is on the first tape and all of the other tapes are blank

Multitape Turing machine computation

When a k-TM starts, its input is on the first tape and all of the other tapes are blank

In each step, the k-TM looks at the current state and the k cells under the k tape heads and then

1. move to a new state;
2. writes k tape symbols, one for each tape; and
3. each head independently moves left or right or stays put

Multitape Turing machine computation

When a k-TM starts, its input is on the first tape and all of the other tapes are blank

In each step, the k-TM looks at the current state and the k cells under the k tape heads and then

1. move to a new state;
2. writes k tape symbols, one for each tape; and
3. each head independently moves left or right or stays put

A k-TM accepts its input if it ever enters the q_{accept} state (same as a normal TM)

Is a k-TM more powerful than a TM?

Natural question: What class of languages do k-TMs recognize?

Is a k-TM more powerful than a TM?

Natural question: What class of languages do k-TMs recognize?

A k-TM can simulate a basic TM by simply ignoring every tape other than its first

Is a k-TM more powerful than a TM?

Natural question: What class of languages do k-TMs recognize?

A k-TM can simulate a basic TM by simply ignoring every tape other than its first

Theorem

Every k-TM can be simulated by a basic TM.

Proof idea

Let's let M be the k-TM and S the single-tape (basic) TM

S needs to keep track of several things:

1. What state M is in
2. What the contents of the k tapes are
3. Where the k tape heads are located

Proof idea

Let's let M be the k-TM and S the single-tape (basic) TM

S needs to keep track of several things:

1. What state M is in
2. What the contents of the k tapes are
3. Where the k tape heads are located

M has only a finite number of states, so S can keep track of them using its own states

For the contents of the tapes, S must use its own tape

For the tape heads, S will use a larger tape alphabet

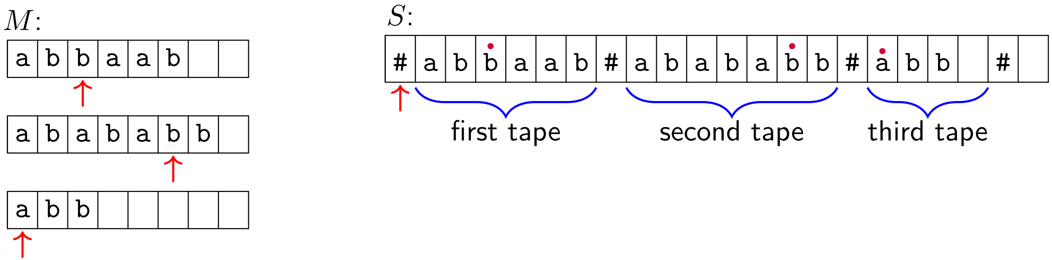
Proof idea continued

If M has tape alphabet Γ, then S will have tape alphabet

$$\Gamma' = \Gamma \cup \{\dot{t} \mid t \in \Gamma\} \cup \{\#\}$$

E.g., if $\Gamma = \{\mathtt{a}, \mathtt{b}, \sqcup\}$, then $\Gamma' = \{\mathtt{a}, \mathtt{b}, \sqcup, \dot{\mathtt{a}}, \dot{\mathtt{b}}, \dot{\sqcup}, \#\}$

The dotted symbols will be used to indicate the position of the tape heads and the # will separate the tapes



Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k + 1$ # symbols and k $\bullet$s

Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k + 1$ # symbols and k $\bullet$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject

Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k+1$ # symbols and k $\dot{}$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject
3. Simulate one move of M by scanning right to see all dotted symbols (using states to remember them)

Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as #$\dot{w}_1 w_2 \cdots w_n$#$\dot{\sqcup}$#$\cdots$#$\dot{\sqcup}$# with $k + 1$ # symbols and k $\bullet$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject
3. Simulate one move of M by scanning right to see all dotted symbols (using states to remember them)
4. Scan back to the first # updating the tape along the way according to M's δ

Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k+1$ # symbols and k $\dot{}$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject
3. Simulate one move of M by scanning right to see all dotted symbols (using states to remember them)
4. Scan back to the first # updating the tape along the way according to M's δ
5. If a tape head moves right onto a #, move the tape to the right and add a $\dot{\sqcup}$.
   What should happen if a head should move left onto a #?

Proof

Proof.
Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k + 1$ # symbols and k $\dot{}$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject
3. Simulate one move of M by scanning right to see all dotted symbols (using states to remember them)
4. Scan back to the first # updating the tape along the way according to M's δ
5. If a tape head moves right onto a #, move the tape to the right and add a $\dot{\sqcup}$.
   What should happen if a head should move left onto a #?
6. Goto 2."

Proof

Proof.

Let M be a k-TM recognizing A. Build a single tape TM S as follows.

S = "On input $w_1 w_2 \cdots w_n$,

1. Format the tape as $\#\dot{w}_1 w_2 \cdots w_n \# \dot{\sqcup} \# \cdots \# \dot{\sqcup} \#$ with $k+1$ # symbols and k $\bullet$s
2. If M is in state q_{accept}, accept, if M is in state q_{reject}, reject
3. Simulate one move of M by scanning right to see all dotted symbols (using states to remember them)
4. Scan back to the first # updating the tape along the way according to M's δ
5. If a tape head moves right onto a #, move the tape to the right and add a $\dot{\sqcup}$.
   What should happen if a head should move left onto a #?
6. Goto 2."

S simulates M and accepts/rejects iff M does so $L(S) = L(M) = A$. □

Nondeterministic Turing machine

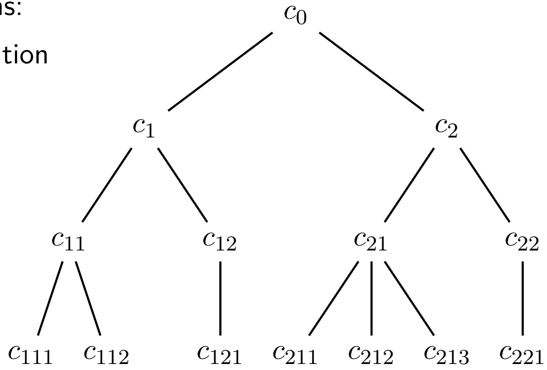
A nondeterministic TM (NTM) is a TM that at each step has zero or more choices

The transition function is what you'd expect: $\delta : Q \times \Gamma \to P(Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\})$

At each step, the NTM looks at its current state and the symbol on the tape under its tape head and then selects the next (state, symbol, direction) from the set of permissible options

Computation is a tree of configurations:

- $c_0 = q_0 w$ is the starting configuration
- Children are the configurations resulting from δ
- w is accepted if any node in the tree is an accepting configuration



Does nondeterminism give more power?

NFAs are no more powerful than DFAs

(Nondeterministic) PDAs *are* more powerful than deterministic PDAs

Are NTMs more powerful than TMs?

Does nondeterminism give more power?

NFAs are no more powerful than DFAs

(Nondeterministic) PDAs *are* more powerful than deterministic PDAs

Are NTMs more powerful than TMs? No

The class of languages recognized by NTMs

Just as a DFA is an NFA that doesn't use nondeterminism, a TM is an NTM that doesn't use nondeterminism

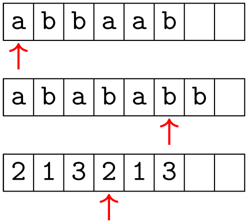
Theorem

Every NTM can be simulated by a TM

Proof idea

We can use a 3-TM to simulate the NTM

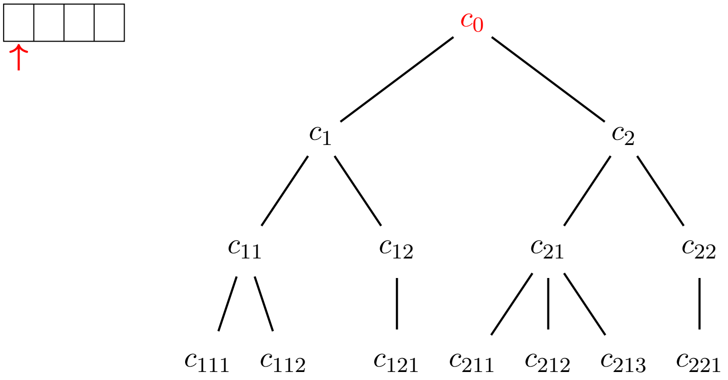
1. The input tape which never changes
2. The work tape will be used to simulate the NTM's tape along a particular nondeterministic branch of computation
3. The address tape identifies the current branch of computation in the tree of configurations



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

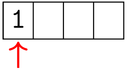
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

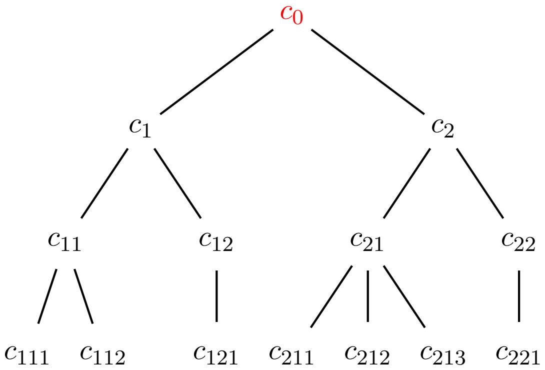


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

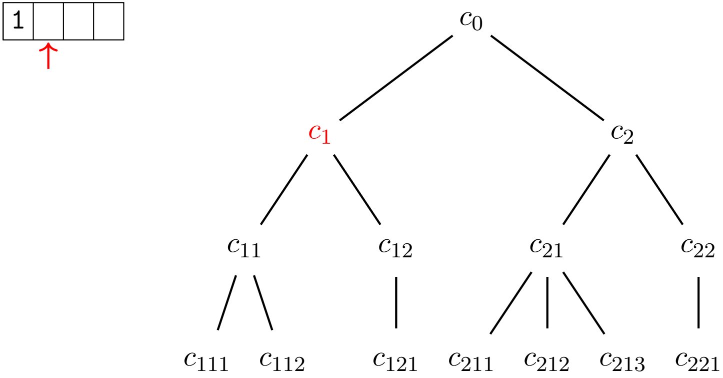




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

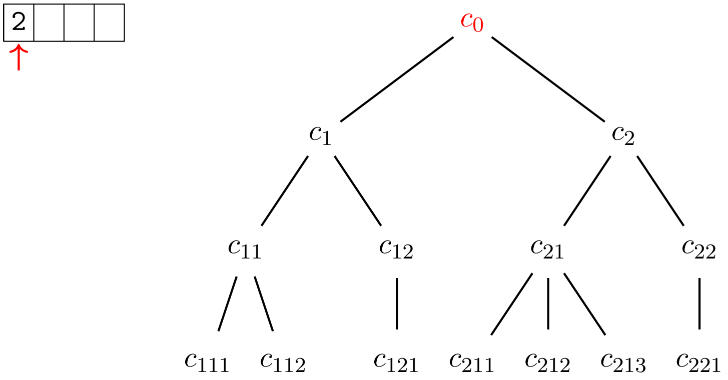
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

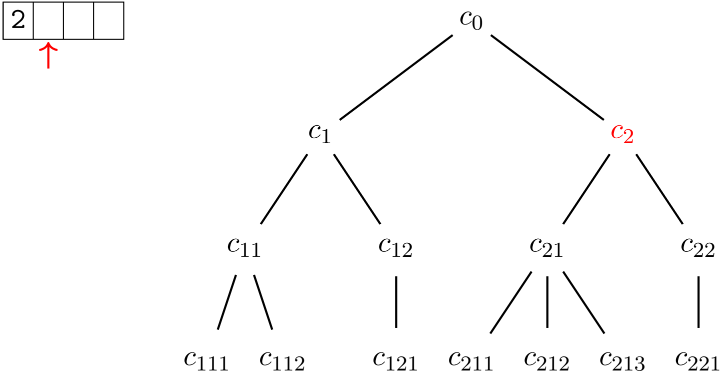
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

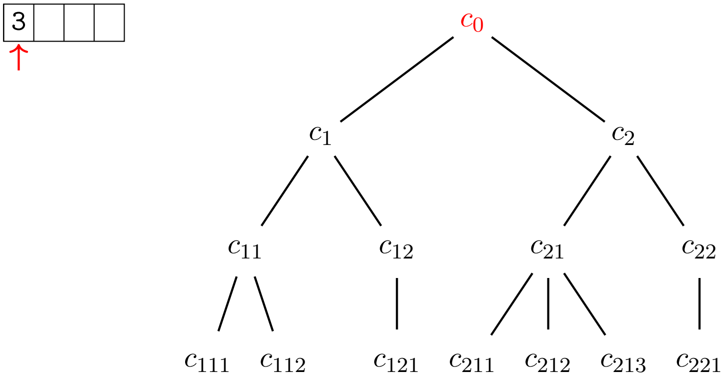
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

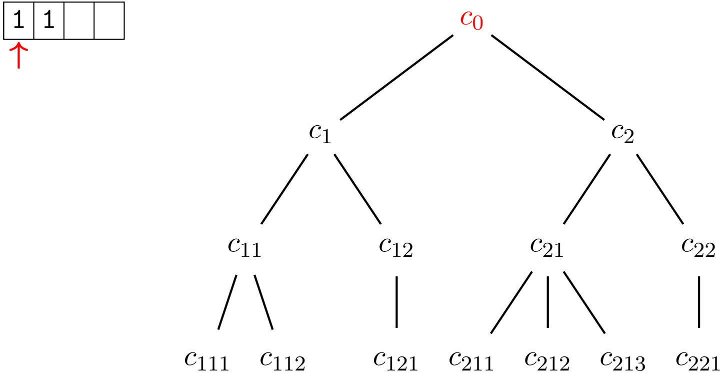
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

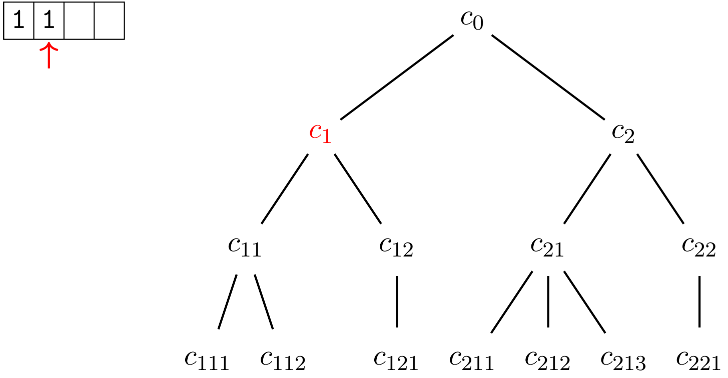
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

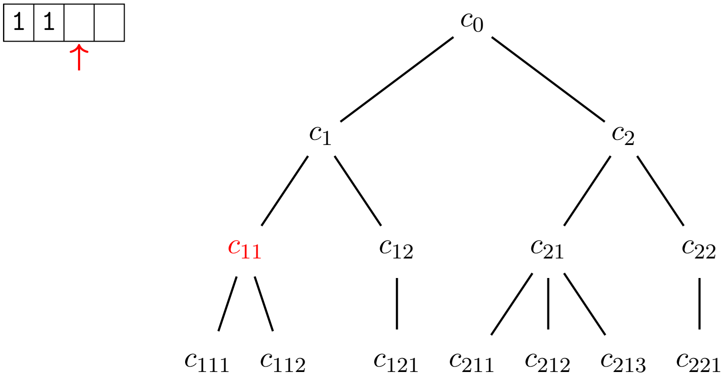
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

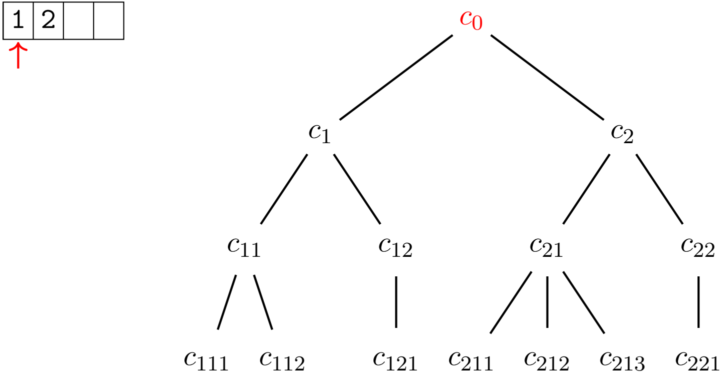
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

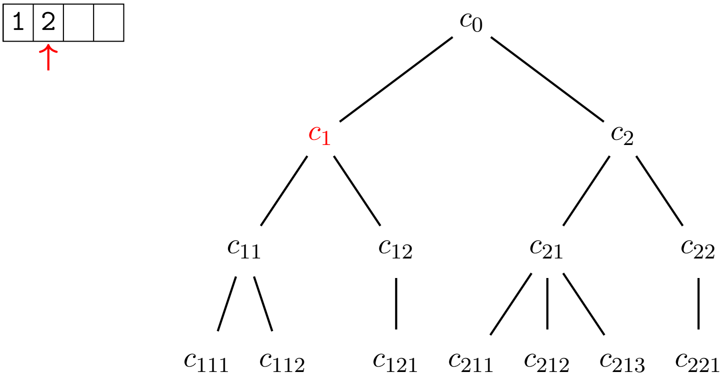
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

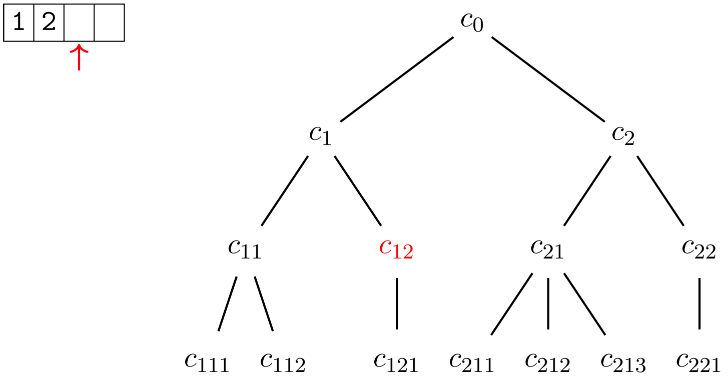
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

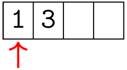
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

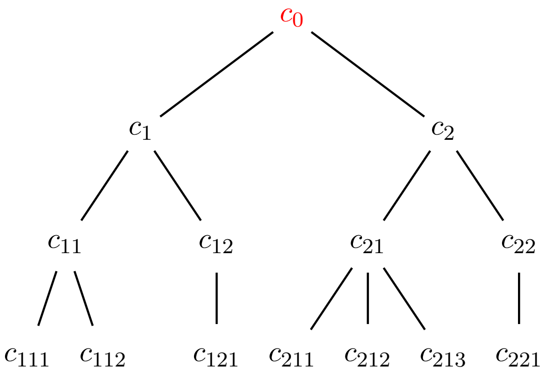


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

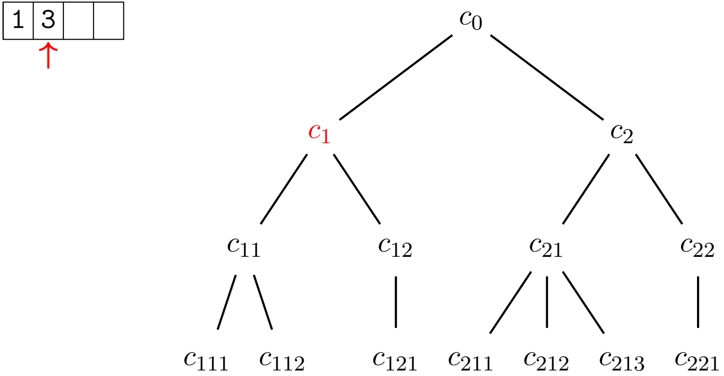




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

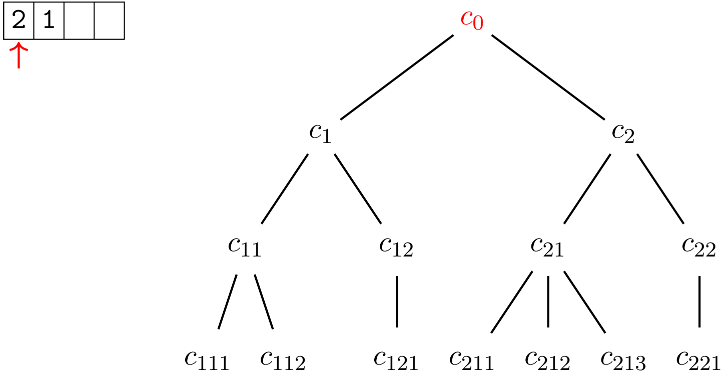
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

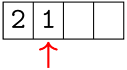
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

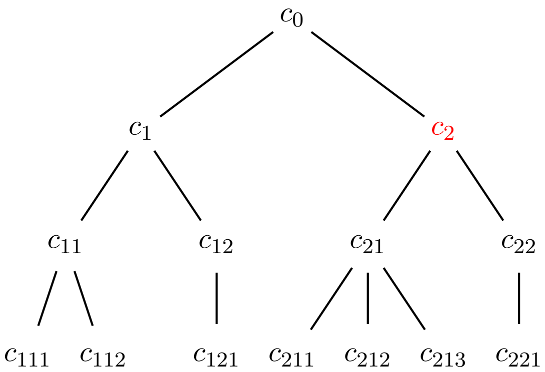


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

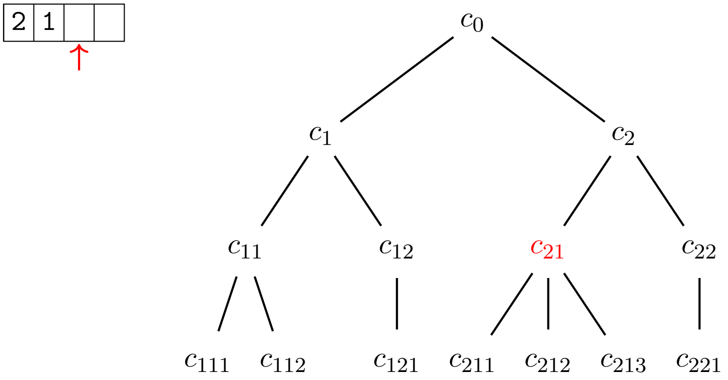




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

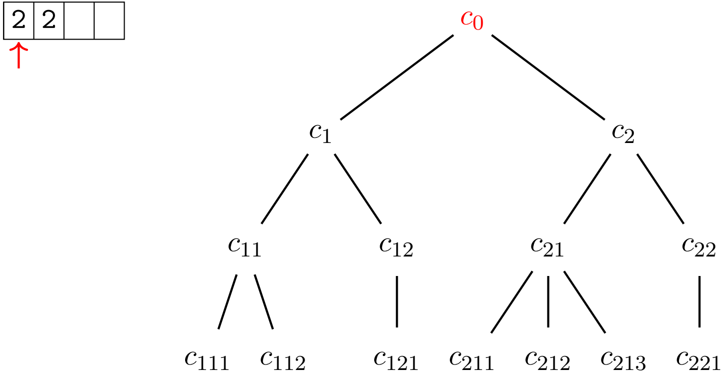
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

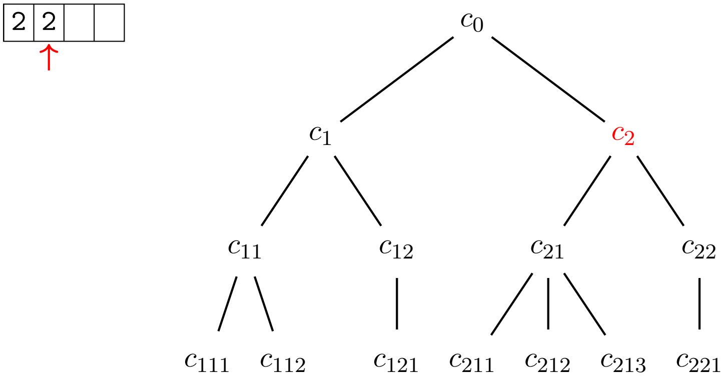
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

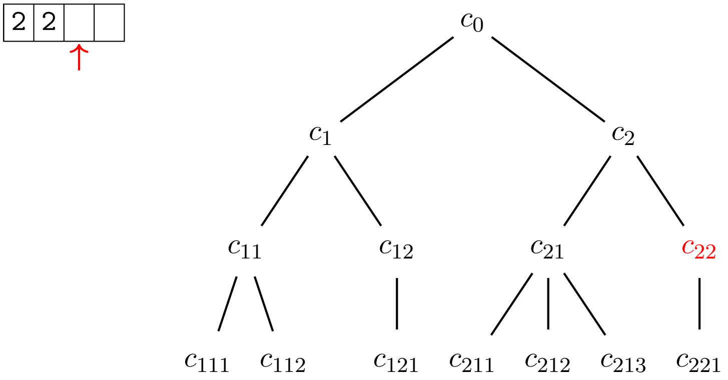
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

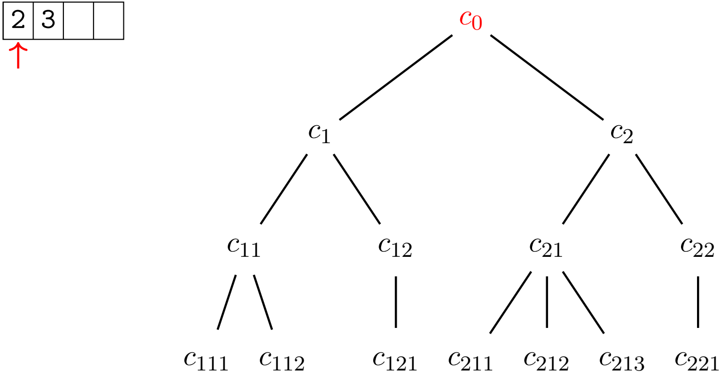
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

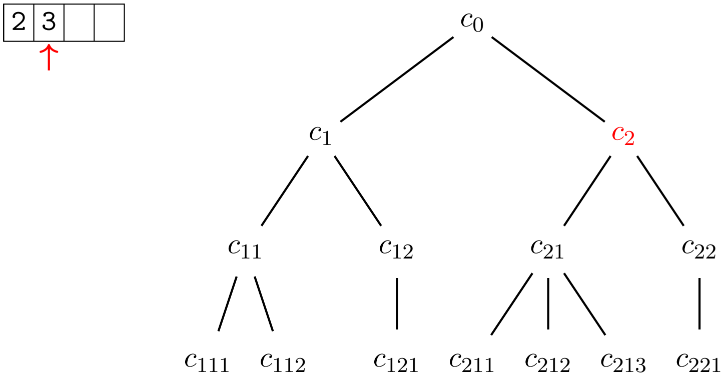
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

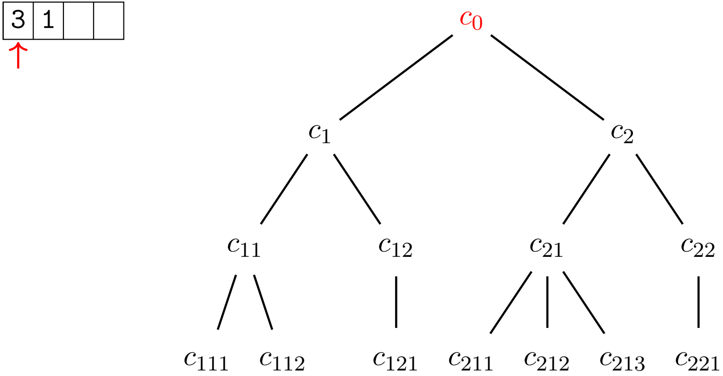
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

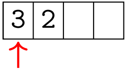
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

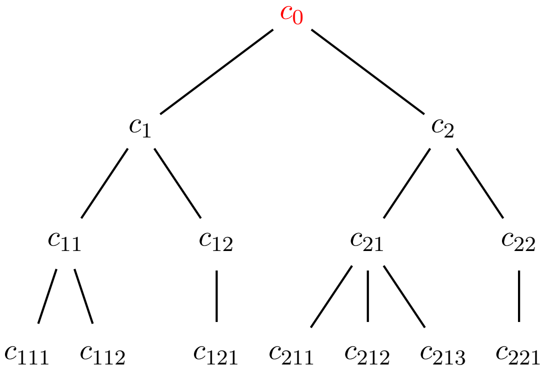
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

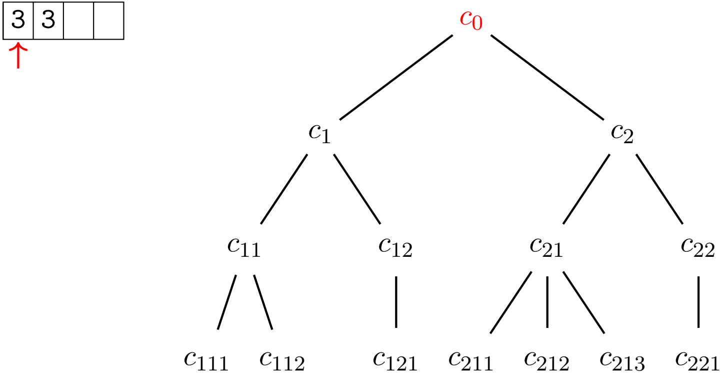




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

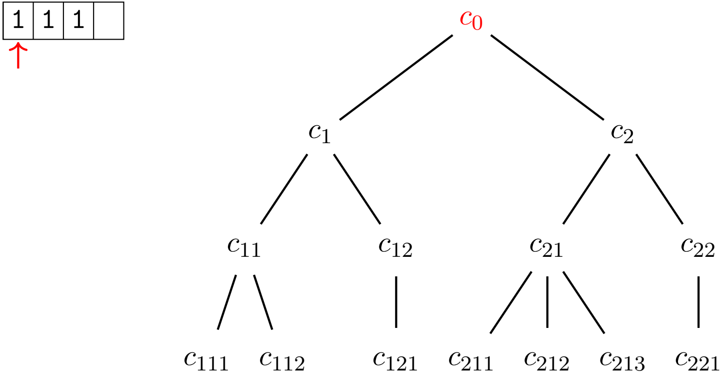
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

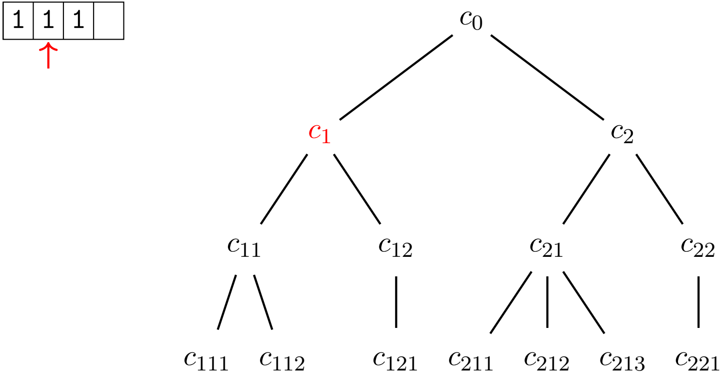
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

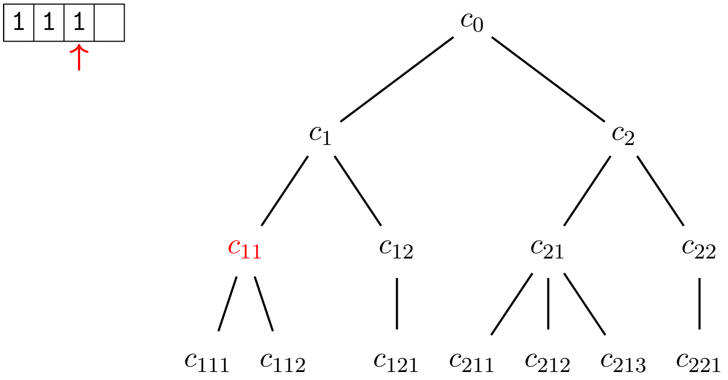
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

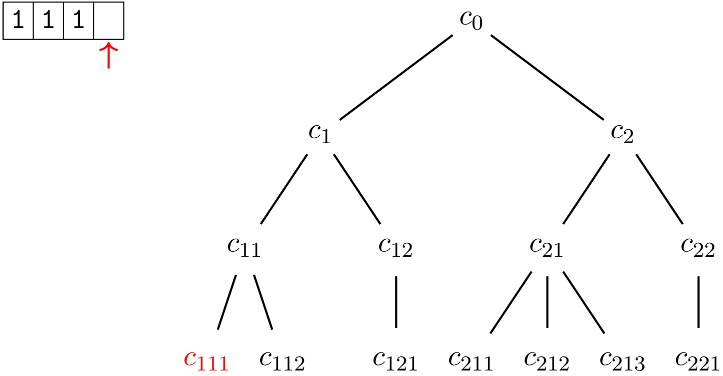
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

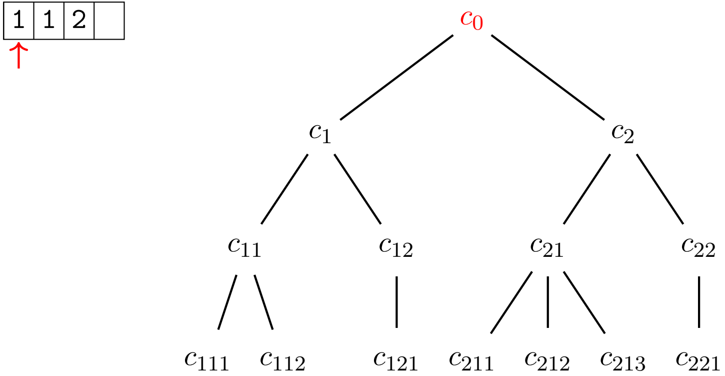
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



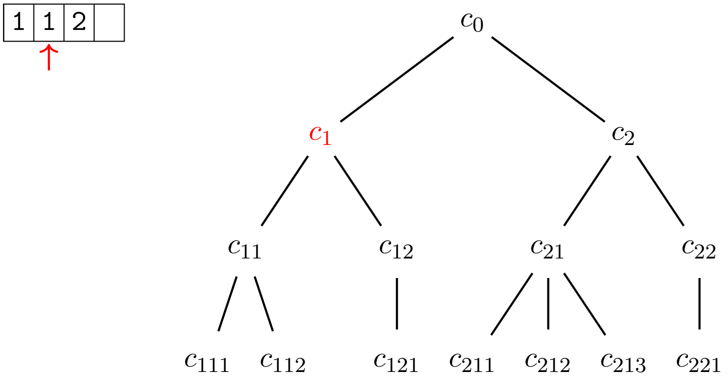
The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

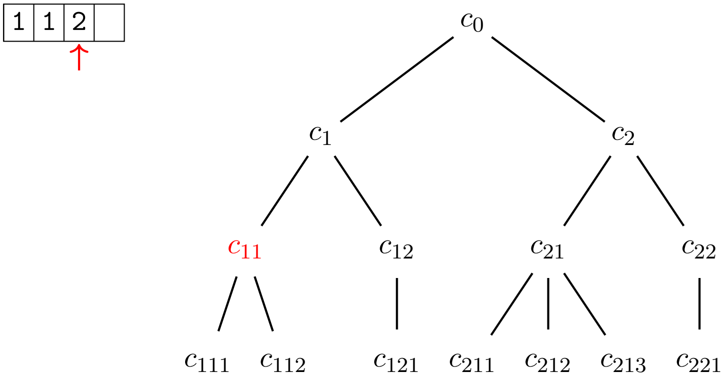
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

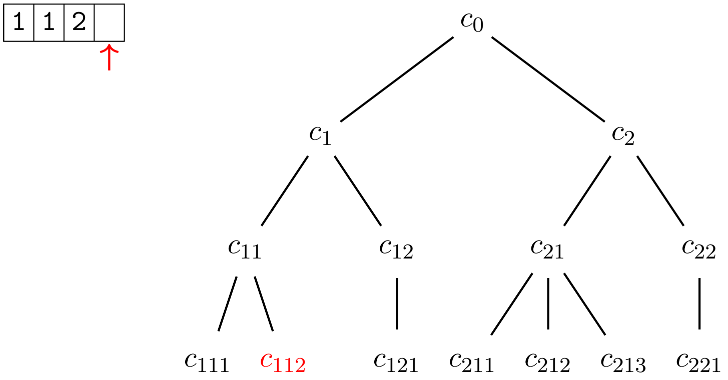
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

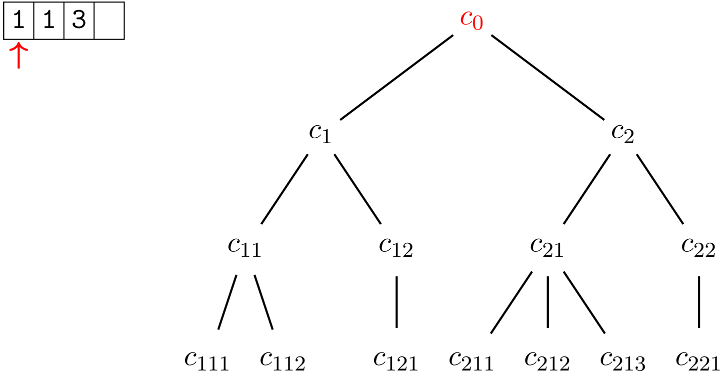
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

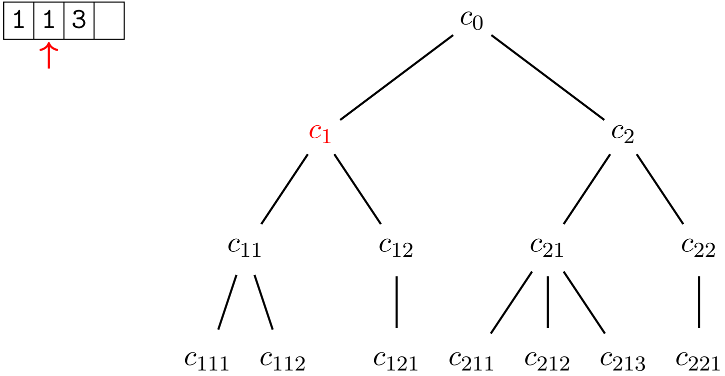
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

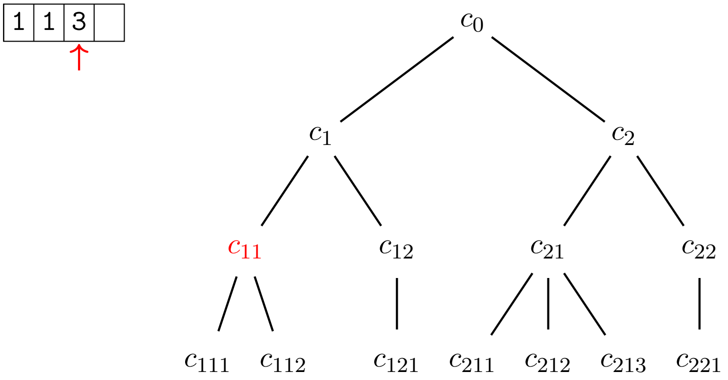
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

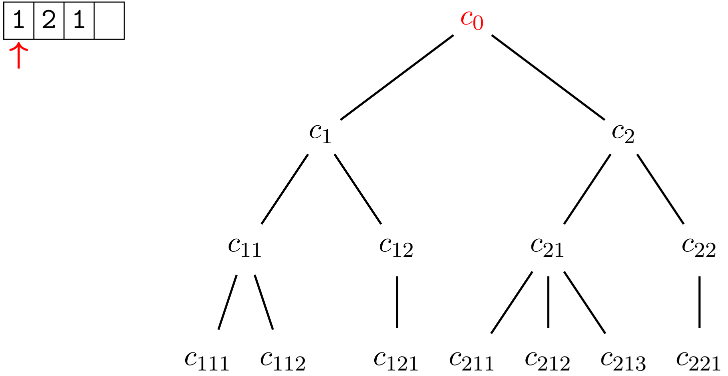
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

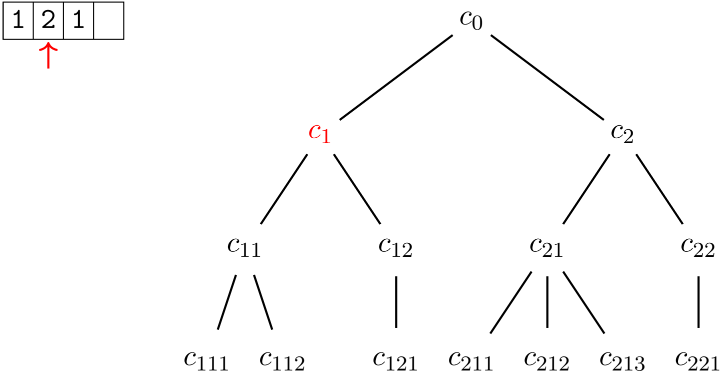
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

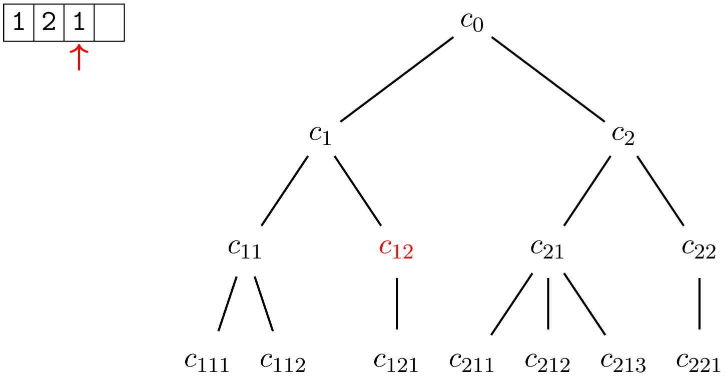
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

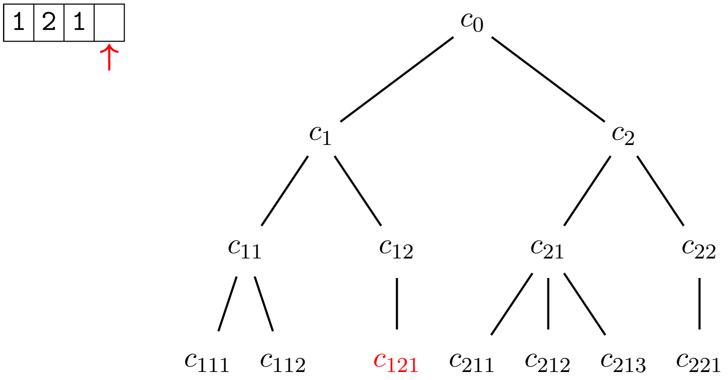
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

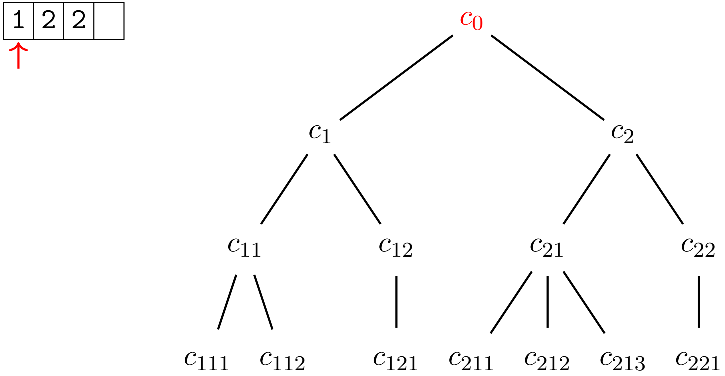
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

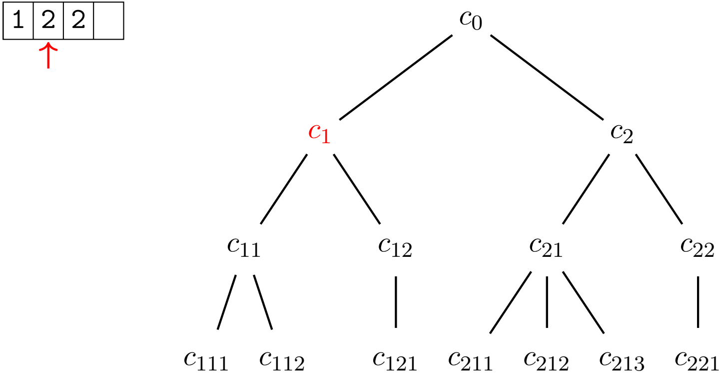
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

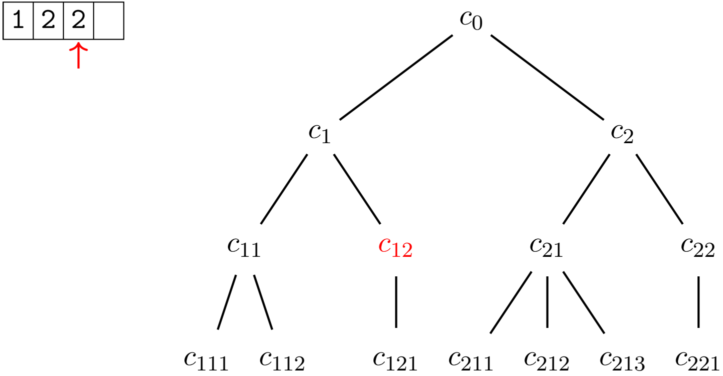
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

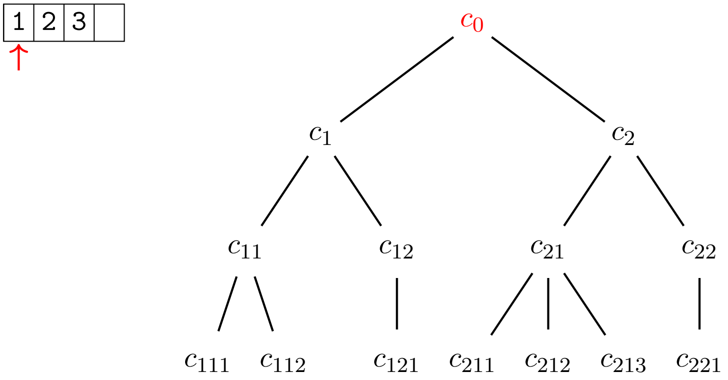
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

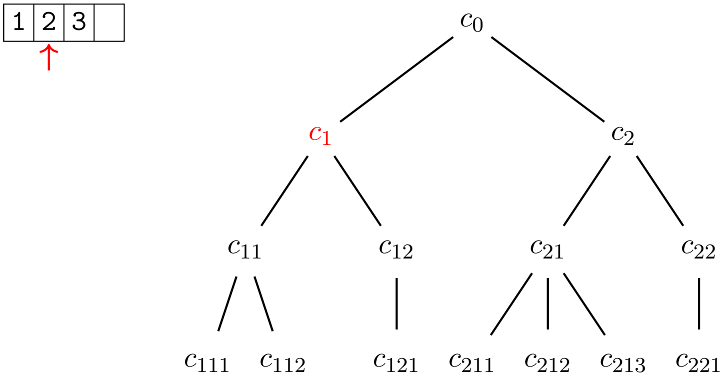
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

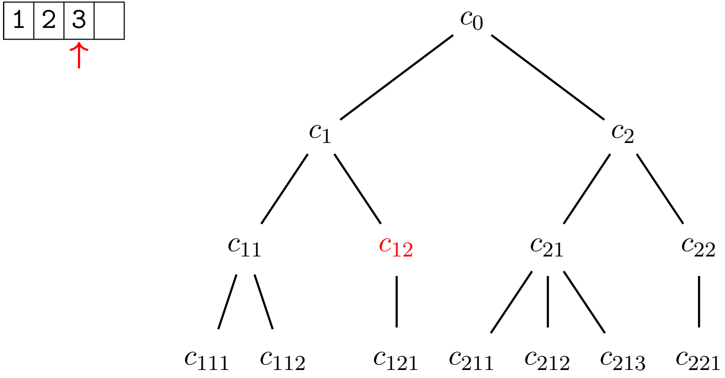
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

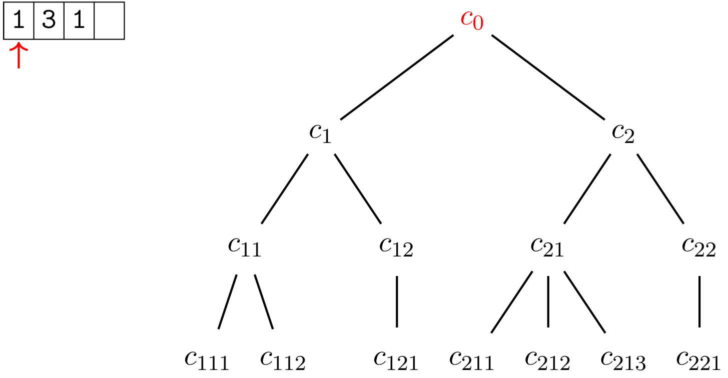
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

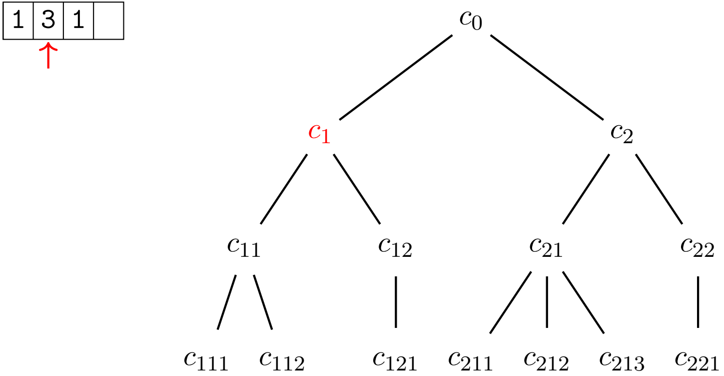
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

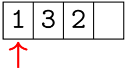
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

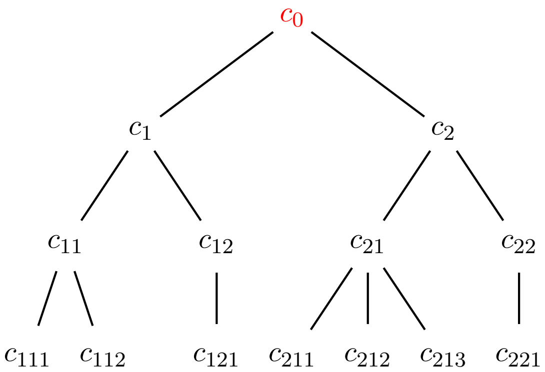


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

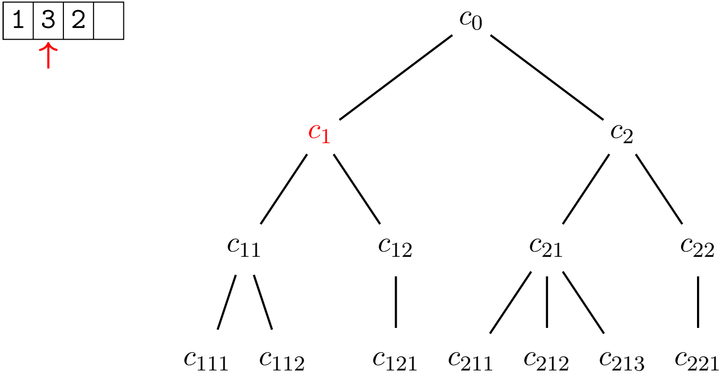




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

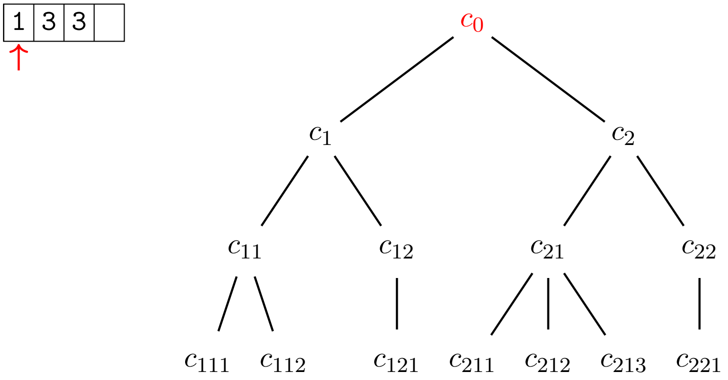
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

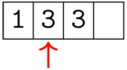
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

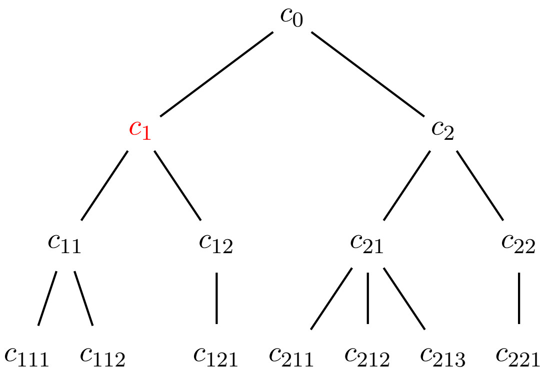


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

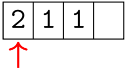


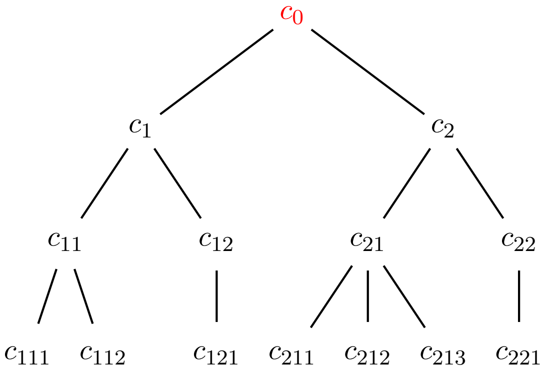


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

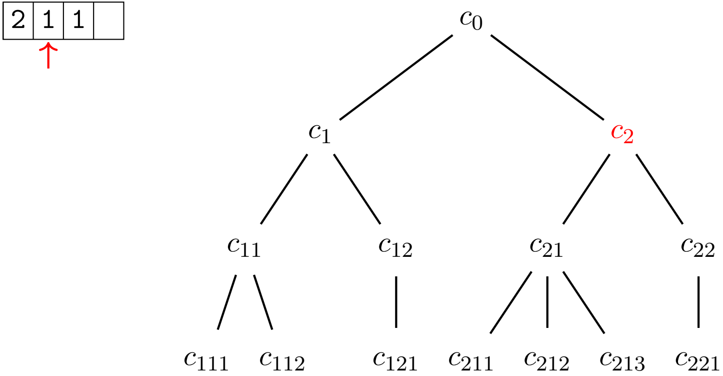




The address tape

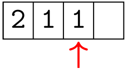
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

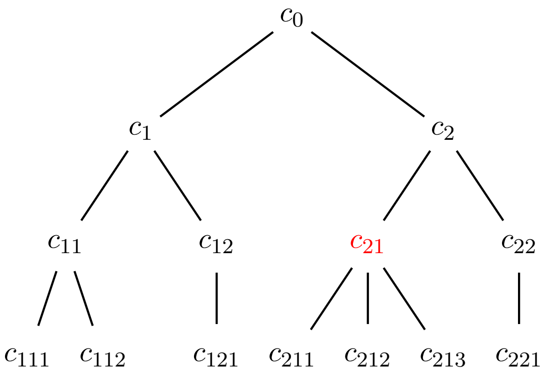
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

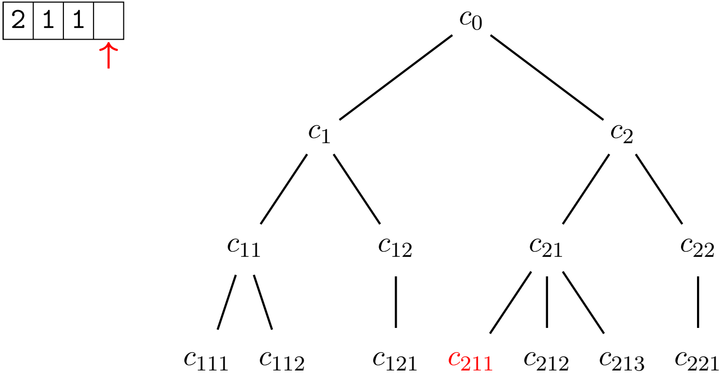




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

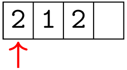
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

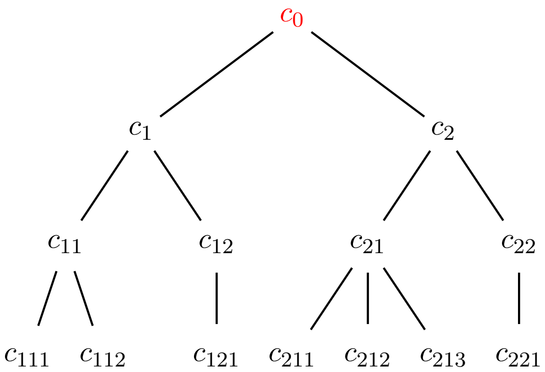


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

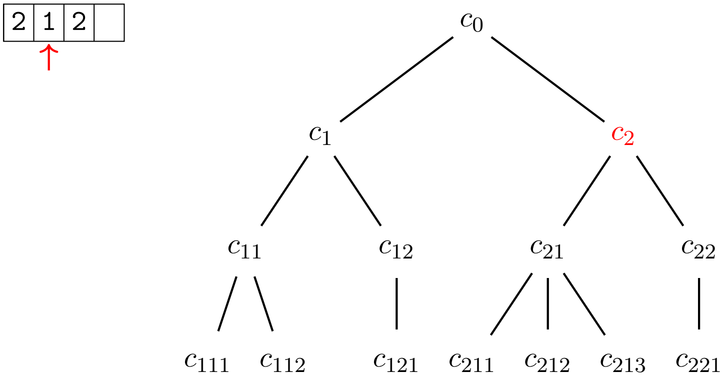




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

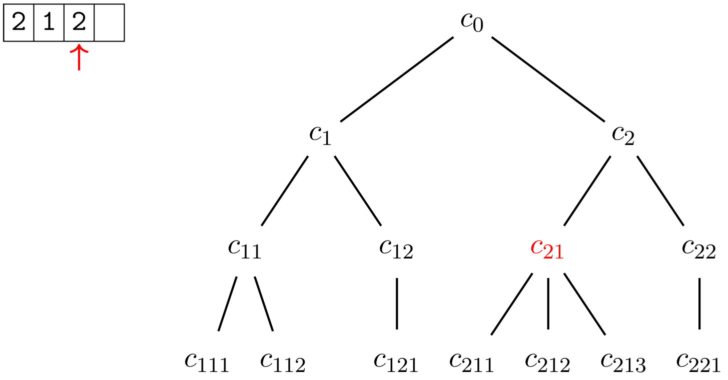
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

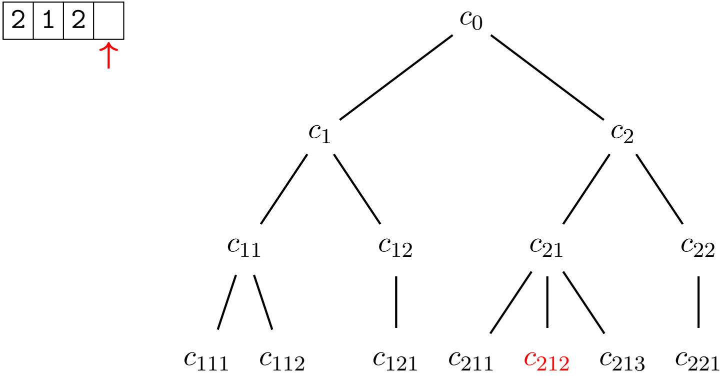
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

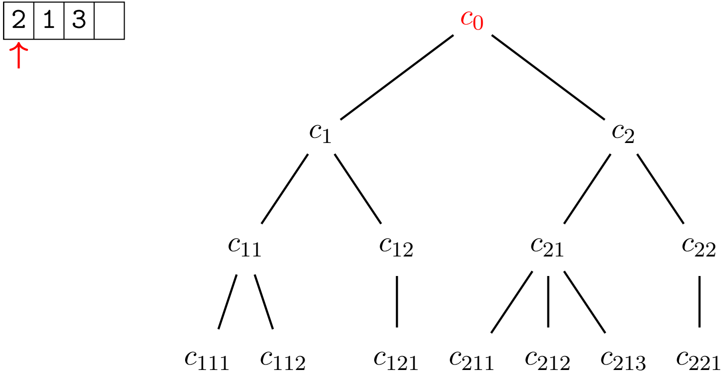
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

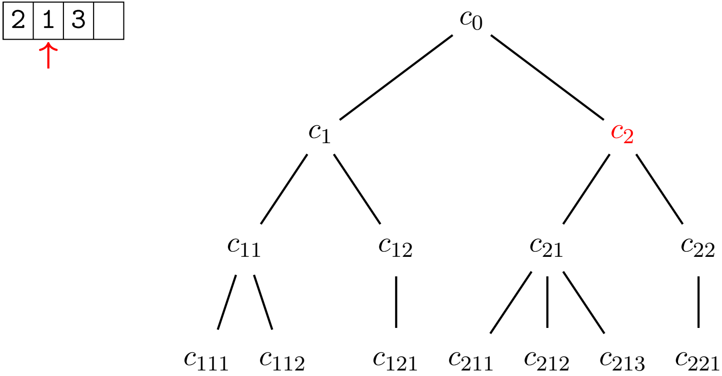
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

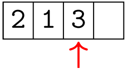
Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

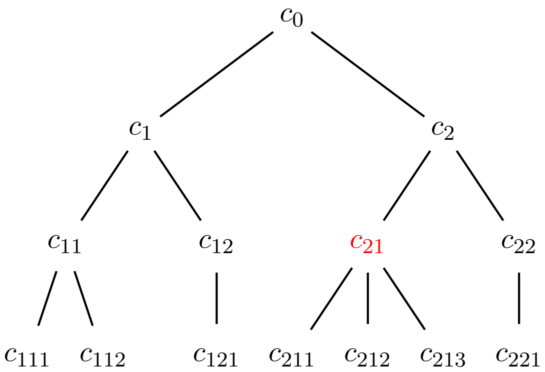


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

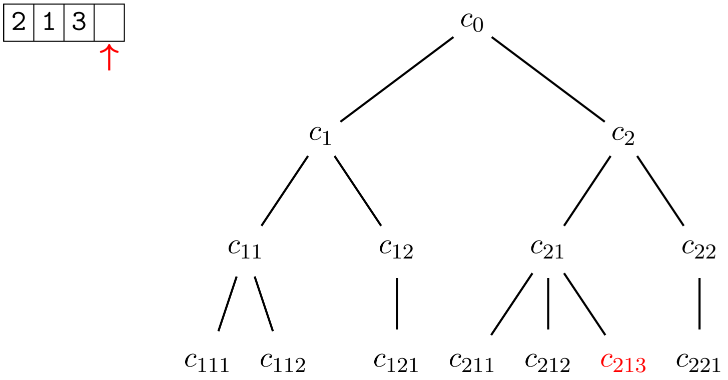




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

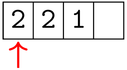
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

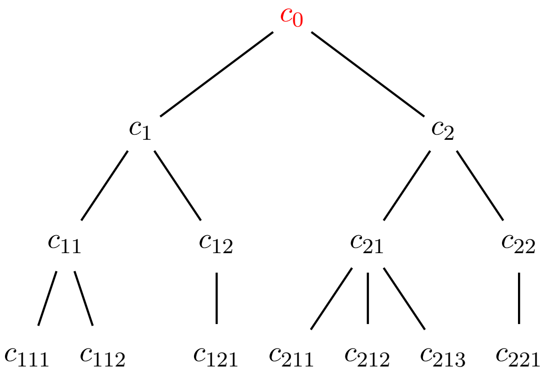


The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape

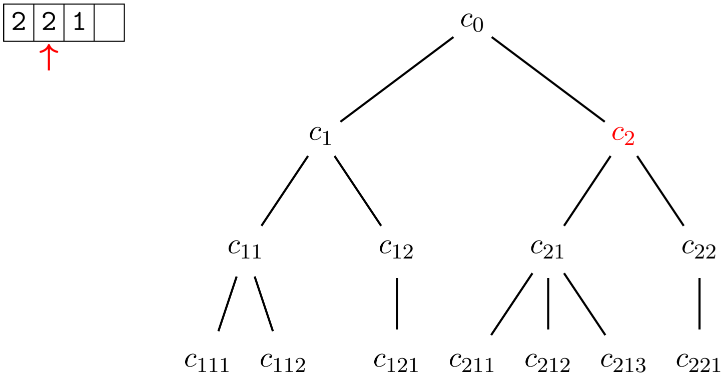




The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

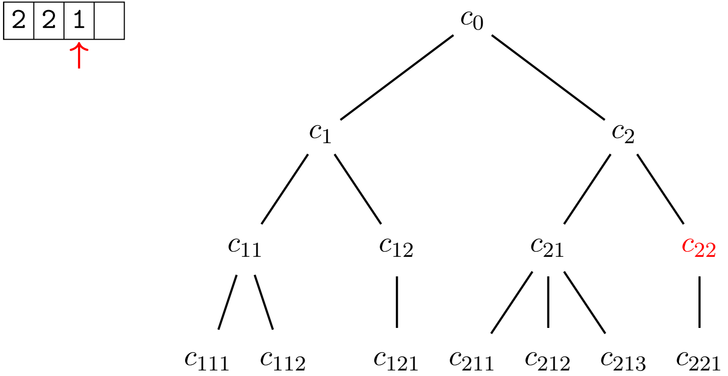
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

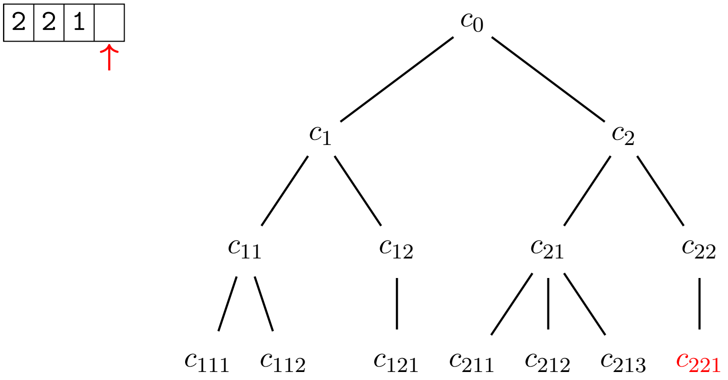
As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



The address tape

Let b be the maximum number of choices the NTM can make at any point, the address tape will be a string over $\{1, 2, \ldots, b\}^*$

As our 3-TM computes, the address tape will select the next node in the tree of configurations to simulate on the work tape



Simulating an NTM N with a 3-TM M

Proof.

M = "On input w

1. Initially, the input tape contains w and the others are empty
2. Copy the input tape to the work tape and return the work tape's head to the beginning of its tape
3. Simulate N on the work tape. For each step, let m be the symbol under the address tape's head. m indicates which of the (at most b) nondeterministic choices to take this step.
4. If N enters an accepting configuration, halt and accept
5. If N enters a rejecting configuration, stop the simulation and goto step 8
6. If the cell under the address tape head is $\sqcup$ or is greater than the actual choices N could make from the current configuration, stop the simulation and goto step 8
7. Otherwise, move to the mth next configuration, move the address tape head to the right one cell and continue the simulation
8. Increment the address tape and goto step 2"

Proof continued

If N accepts w, then some tree in the configuration is an accepting configuration

M performs a breadth-first search through the tree of configurations so it will eventually find the accepting configuration and accept

If N rejects w or loops on w, then M will loop

Thus, $L(M) = L(N)$. $\square$

Alternative characterizations of Turing-recognizable languages

The following statements about a language A are equivalent

- A is Turing-recognizable
- A is recognized by some TM
- A is recognized by some TM with stay put
- A is recognized by some k-TM
- A is recognized by some NTM

Alternative characterization of decidable languages

If you look closely at the proofs of equivalence of TMs, TMs with stay put, and k-TMs, you see that if the original (fancy) TM is a decider, then the (basic) TM we built is too

Alternative characterization of decidable languages

If you look closely at the proofs of equivalence of TMs, TMs with stay put, and k-TMs, you see that if the original (fancy) TM is a decider, then the (basic) TM we built is too

But the proof for NTMs was different: If N rejected even a single string, then the M we built is not a decider

Alternative characterization of decidable languages

If you look closely at the proofs of equivalence of TMs, TMs with stay put, and k-TMs, you see that if the original (fancy) TM is a decider, then the (basic) TM we built is too

But the proof for NTMs was different: If N rejected even a single string, then the M we built is not a decider

We can actually fix that by keeping track of whether every step at a particular depth of the configuration tree was invalid or lead to a rejecting configuration

If they were all rejecting or invalid, then the NTM halted on all branches of the computation on w so M should halt too

We just need to modify the TM to keep track of whether a particular depth had an address that was valid and non-rejecting and check that before moving to the next depth

Alternative characterizations of decidable languages

The following statements about a language A are equivalent

- A is decidable
- A is decided by some TM
- A is decided by some TM with stay put
- A is decided by some k-TM
- A is decided by some NTM

Next time

We've reached the point in the course where it is no longer useful to talk about low level details of TMs

We're going to start talking about *algorithms* and the limits of computation

Everything in the course so far was building up to this point (although they are interesting topics in their own right!)