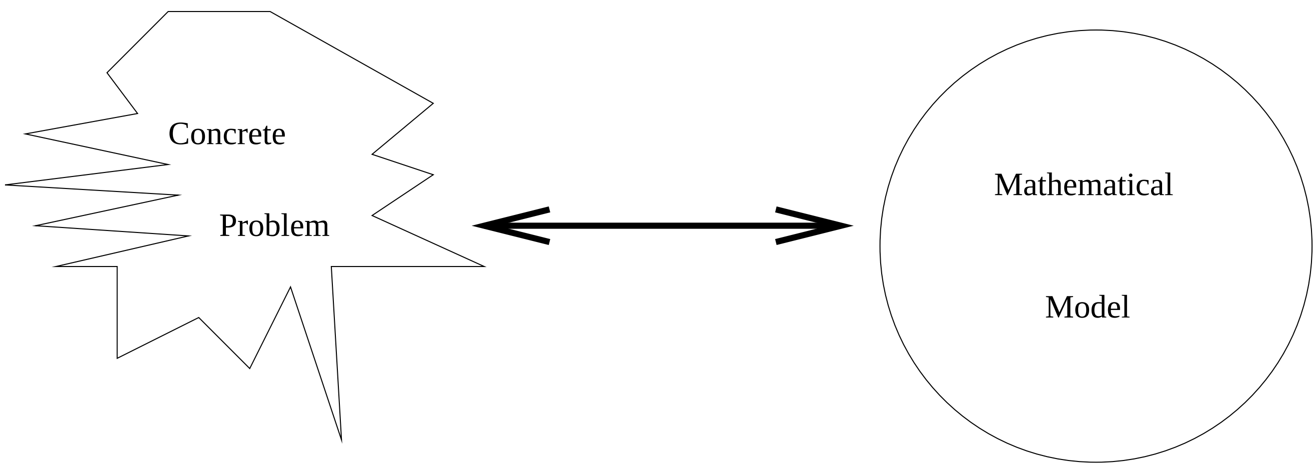
In-depth introduction to main models, concepts of theory of computation:

- **Computability**: what can be computed in principle
- **Logic**: how can we express our requirements
- **Complexity**: what can be computed in practice



Formal Models of Computation:

- Finite-state
- Stacks = CFL
- Turing Machine
- Logical Formula

Texts: available at Jeffery Amherst College Store

[P]: Christos Papadimitriou, *Computational Complexity*

[BE:] Jon Barwise and John Etchemendy, *Language, Proof, and Logic*

Prerequisites: Mathematical maturity: reason abstractly, understand and write proofs. CMPSCI 250 needed; CMPSCI 311, 401 helpful. Today's material is a good taste of the sort of stuff we will do.

Work:

- eight problem sets (35% of grade)
- midterm (30% of grade)
- final (35% of grade)

Cooperation: Students should talk to each other and help each other; but **write up solutions on your own, in your own words.** Sharing or copying a solution could result in failure. If a significant part of one of your solutions is due to someone else, or something you've read then **you must acknowledge your source!**

Mathematical Sophistication

- *How to Read and Do Proofs, Second Edition* by Daniel Solow, 1990, John Wiley and Sons.

Review of Regular and Context-Free Languages

- Hopcroft, Motwani, and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2001: Chapters 1–6.
- Lewis and Papadimitriou, *Elements of the Theory of Computation*, 1998: Chapters 1–3.
- Sipser, *Introduction to the Theory of Computation*, 1997: Chapters 1 – 2.

NP Completeness

- Garey and Johnson, *Computers and Intractability*, 1979.

Descriptive Complexity

- Immerman, *Descriptive Complexity*, 1999.

Syllabus will be up soon on the course web site:

- `http://www.cs.umass.edu/ barring/cs601`

There is a pointer there to the Spring 2002 web site, and the syllabus there will be close to what we do here.

Rough guide:

- Formal Languages and Computability (9 lectures)
- Propositional and First-Order Logic (7 lectures)
- Complexity Theory (11 lectures)

Definition: An **alphabet** is a non-empty finite set, e.g., $\Sigma = \{0, 1\}$, $\Gamma = \{a, b, c\}$, etc.

Definition: A **string** over an alphabet Σ is a finite sequence of zero or more symbols from Σ. The unique string with zero symbols is called ϵ. The set of all strings over Σ is called Σ^*.

Definition: A **language** over Σ is any subset of Σ^*. The decision problem for a language L is to input a string w and determine whether $w \in L$.

Definition: The set of **regular expressions** $R(\Sigma)$ over alphabet Σ is the smallest set of strings such that:

1. if $a \in \Sigma$ then $a \in R(\Sigma)$
2. $\epsilon \in \mathbf{R}(\Sigma)$
3. $\emptyset \in \mathbf{R}(\Sigma)$
4. if $e, f \in R(\Sigma)$ then so are the following:
   (a) $(e \cup f)$
   (b) $(e \circ f)$
   (c) $(e^\star)$

Examples:

- $e_1 = 0^\star \in R(\{0,1\})$
- $e_2 = ((a \cup b) \circ (a \cup b))^\star \in R(\{a,b\})$
- $e_3 = a^\star(ba^\star ba^\star)^\star \in R(\{a,b,c\})$

Meanings:

- $\mathcal{L}(0^\star) \;=\; \{\epsilon, 0, 00, 0^3, 0^4, \ldots\} \;=\; \{0^i \mid i \in \mathbf{N}\}$
- $\mathcal{L}((a \cup b)^{2\,\star}) \quad = \quad \{w \in \{a,b\}^\star \mid |w| \equiv 0 \pmod 2\}$
- $\mathcal{L}(a^\star(ba^\star ba^\star)^\star) \quad = \quad \{w \in \{a,b\}^\star \mid \#_b(w) \equiv 0 \pmod 2\}$

Recall the meaning of Kleene star, for any set, A,

$$\begin{aligned}
A^\star &\equiv \bigcup_{i=0}^{\infty} A^i \\
&\equiv A^0 \cup A^1 \cup A^2 \cup \cdots \\
&\equiv \{\epsilon\} \cup A \cup \{xy \mid x, y \in A\} \cup \cdots \\
&\equiv \{x_1 x_2 \ldots x_n \mid n \in \mathbf{N}; x_1, \ldots, x_n \in A\}
\end{aligned}$$

Meaning of a Regular Expression:

1. if $a \in \Sigma$ then $a \in R(\Sigma)$; $\mathcal{L}(a) = \{a\}$
2. $\epsilon \in \mathbf{R}(\Sigma)$; $\mathcal{L}(\epsilon) = \{\epsilon\}$
3. $\emptyset \in \mathbf{R}(\Sigma)$; $\mathcal{L}(\emptyset) = \emptyset$
4. if $e, f \in R(\Sigma)$ then so are $(e \cup f)$, $(e \circ f)$, $(e^\star)$:

$$\begin{aligned}
\mathcal{L}(e \cup f) &= \mathcal{L}(e) \cup \mathcal{L}(f) \\
\mathcal{L}(e \circ f) &= \mathcal{L}(e)\mathcal{L}(f) = \{uv \mid u \in \mathcal{L}(e), v \in \mathcal{L}(f)\} \\
\mathcal{L}(e^\star) &= (\mathcal{L}(e))^\star
\end{aligned}$$

Definition 1.1 $A \subseteq \Sigma^\star$ is **regular** iff

$$(\exists e \in R(\Sigma))(A = \mathcal{L}(e))$$

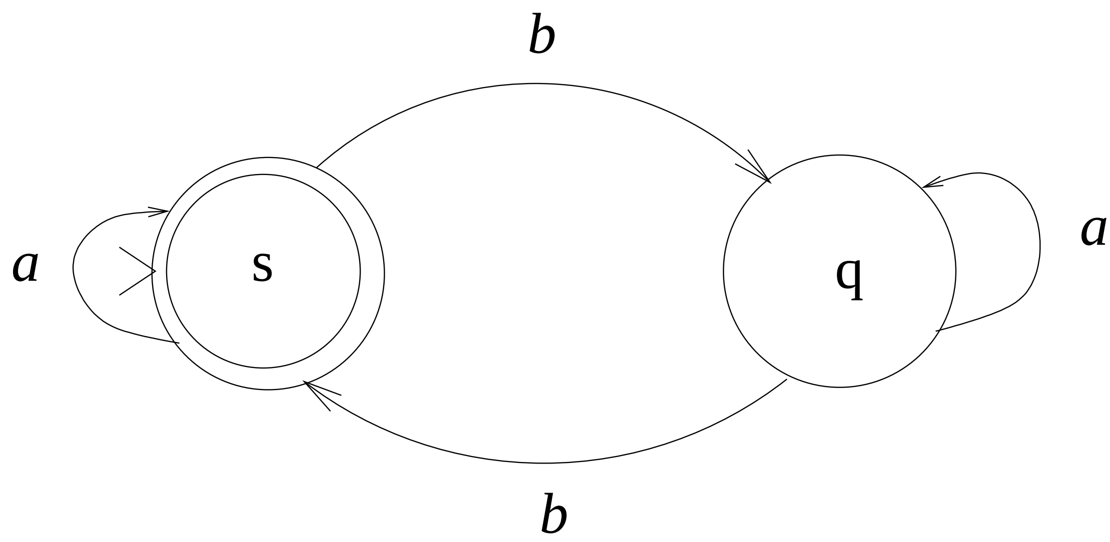
In other words, a set, A, is regular iff there exists a regular expression that denotes it.

Definition: A **deterministic finite automaton (DFA)** is a tuple,

$$D = (Q, \Sigma, \delta, s, F)$$

- Q is a finite set of states,
- Σ is a finite alphabet,
- $\delta : Q \times \Sigma \to Q$ is the transition function,
- $s \in Q$ is the start state, and
- $F \subseteq Q$ is the set of final or accept states.

$$D_1 \;=\; (\{s,q\},\{a,b\},\delta_1,s,\{s\})$$
$$\delta_1 \;=\; \{\langle\langle s,a\rangle,s\rangle,\langle\langle s,b\rangle,q\rangle,\langle\langle q,a\rangle,q\rangle,\langle\langle q,b\rangle,s\rangle\}$$



$a \quad a \quad b \quad b \quad a \quad b \quad a \quad a \quad b \quad a$

s

$$\mathcal{L}_1 \;=\; \mathcal{L}(D_1) \;=\; \{w \in \{a,b\}^\star \;\mid\; \#_b(w) \equiv 0 \pmod 2\}$$

$$\mathcal{L}_1 \quad = \quad \mathcal{L}(a^\star(ba^\star ba^\star)^\star)$$

Definition: A **nondeterministic finite automaton (NFA)** is a tuple,

$$N = (Q, \Sigma, \Delta, s, F)$$

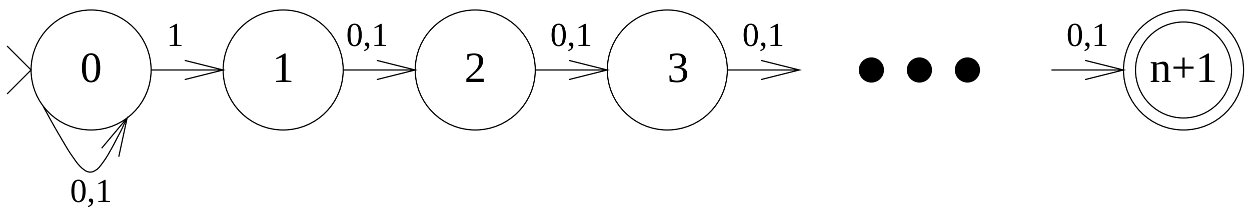
- Q is a finite set of states,
- Σ is a finite alphabet,
- $\Delta : (Q \times (\Sigma \cup \{\epsilon\}) \to \wp(Q)$ is the transition function,
- $s \in Q$ is the start state, and
- $F \subseteq Q$ is the set of final or accept states.

$$\mathcal{L}(N) \quad = \quad \{w \mid s \underset{w}{\overset{\star}{\to}} q \in F\}$$

$$\wp(S) \quad = \quad \text{power set of } S \quad = \quad \{A \mid A \subseteq S\}$$

$$N_n = (\{q_0, \ldots, q_{n+1}\}, \{0,1\}, \Delta_n, q_0, \{q_{n+1}\})$$

$$\Delta_n = \{\langle\langle q_0, 0\rangle, \{q_0\}\rangle, \langle\langle q_0, 1\rangle, \{q_0, q_1\}\rangle, \ldots, \langle\langle q_n, 1\rangle, \{q_{n+1}\}\rangle\}$$

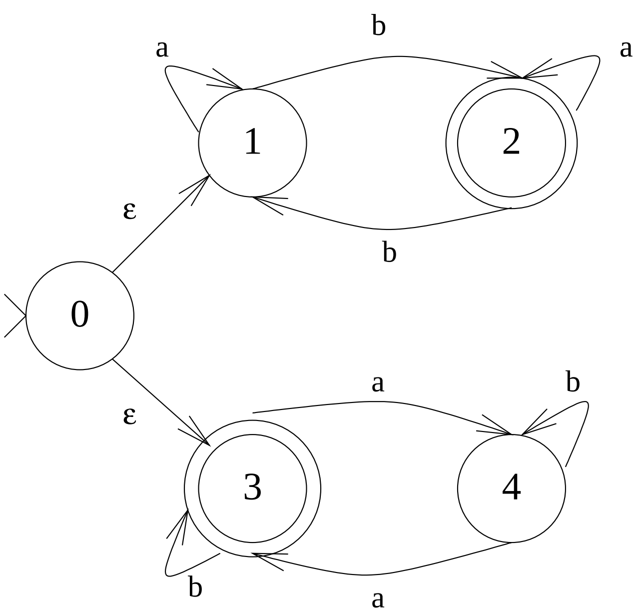


[You will show in HW 1 that to accept $\mathcal{L}(N_n)$, a DFA would need 2^{n+1} states.]

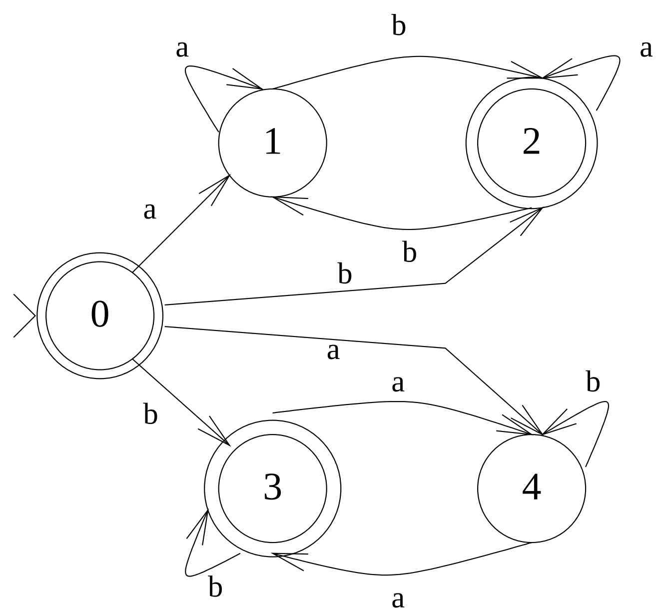
Proposition 1.2 *Every NFA N can be translated into an NFA wo ϵ-transitions N' s.t. $\mathcal{L}(N) = \mathcal{L}(N')$*

Proof: Given $N = (Q, \Sigma, \Delta, q_0, F)$, let $N' = (Q, \Sigma, \Delta', q_0, F')$ where

$$\begin{aligned} \Delta'(q, a) \; &= \; \{r \mid (\exists s, t) q \xrightarrow{\epsilon^\star} s \xrightarrow{a} t \xrightarrow{\epsilon^\star} r\} \\ \mathbf{F}' \; &= \; \{q \mid (\exists s \in F) q \xrightarrow{\epsilon^\star} s\} \end{aligned}$$



N



N' ♠

Notation: For a DFA, $D = (Q, \Sigma, \delta, s, F)$, let $\delta^\star(q, w)$ be the state that D will be in after reading string w, when started in q,

$$\delta^\star(q, \epsilon) \equiv q$$
$$\delta^\star(q, wa) \equiv \delta(\delta^\star(q, w), a)$$

$$\mathcal{L}(D) \equiv \{w \mid \delta^\star(s, w) \in F\}$$

For an NFA without ϵ transitions, $N = (Q, \Sigma, \Delta, s, F)$, let $\Delta^\star(q, w)$ be the set of states that N can be in after reading string w, when started in q,

$$\begin{aligned} \Delta^\star(q, \epsilon) &:= \{q\} \\ \Delta^\star(q, wa) &:= \bigcup_{r \in \Delta^\star(q,w)} \Delta(r, a) \end{aligned}$$

$$\mathcal{L}(N) \equiv \{w \mid \Delta^\star(s, w) \cap F \neq \emptyset\}$$

Proposition 1.3 *For every NFA, N, with n states, there is a DFA, D, with at most 2^n states s.t. $\mathcal{L}(D) = \mathcal{L}(N)$.*

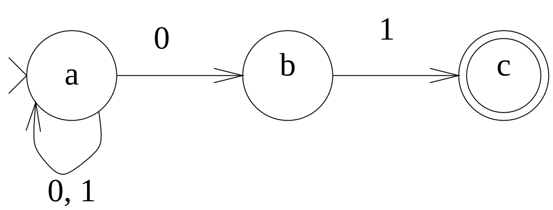
Proof: Let $N = (Q, \Sigma, \Delta, q_0, F)$. By Proposition 1.2 may assume that N has no ϵ transitions.

Let $D = (\wp(Q), \Sigma, \delta, \{q_0\}, F')$

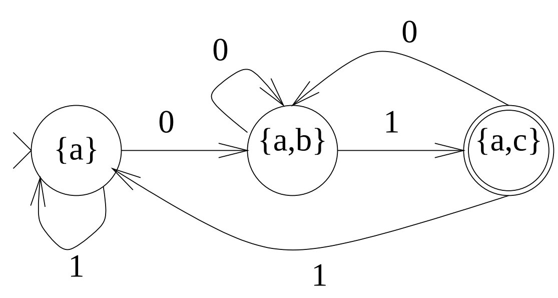
$$\delta(S, a) = \bigcup_{r \in S} \Delta(r, a)$$

$$F' = \{S \subseteq Q \mid S \cap F \neq \emptyset\}$$

N



D



Claim: For all $w \in \Sigma^\star$,

$$\delta^\star(\{q_0\}, w) \quad = \quad \Delta^\star(q_0, w)$$

By induction on $|w|$:

$|w| = 0$: $\delta^\star(\{q_0\}, \epsilon) \ = \ \{q_0\} \ = \ \Delta^\star(q_0, \epsilon)$

$|w| = k + 1$: $w = ua$.

Inductively, $\delta^\star(\{q_0\}, u) \ = \ \Delta^\star(q_0, u)$

$$\begin{aligned} \delta^\star(\{q_0\}, ua) &= \delta(\delta^\star(\{q_0\}, u), a) \\ &= \bigcup_{r \in \delta^\star(\{q_0\}, u)} \Delta(r, a) \\ &= \bigcup_{r \in \Delta^\star(q_0, u)} \Delta(r, a) \\ &= \Delta^\star(q, ua) \end{aligned}$$

Therefore, $\mathcal{L}(D) = \mathcal{L}(N)$.

Theorem 1.4 (Kleene's Th) *Let $A \subseteq \Sigma^\star$ be any language. Then the following are equivalent:*

1. *$A = \mathcal{L}(D)$, for some DFA D.*
2. *$A = \mathcal{L}(N)$, for some NFA N wo ϵ transitions*
3. *$A = \mathcal{L}(N)$, for some NFA N.*
4. *$A = \mathcal{L}(e)$, for some regular expression e.*
5. *A is regular.*

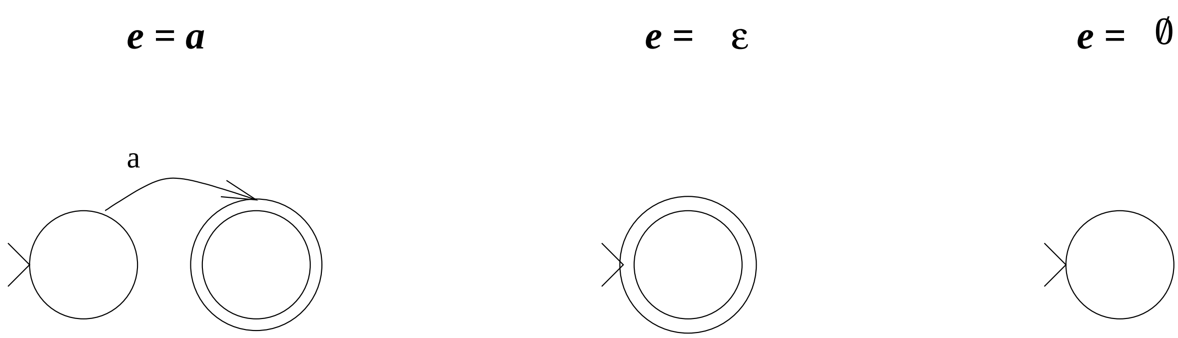
Proof: Obvious that $1 \to 2 \to 3$.

$3 \to 2$ by Prop. 1.2.

$2 \to 1$ by Prop. 1.3 (subset construction).

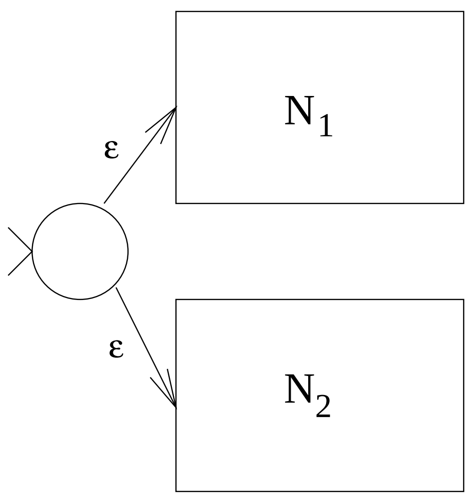
$4 \leftrightarrow 5$ by def of regular

$4 \to 3$: We show by induction on the number of symbols in the regular expression e, that there is an NFA N with $\mathcal{L}(e) = \mathcal{L}(N)$:



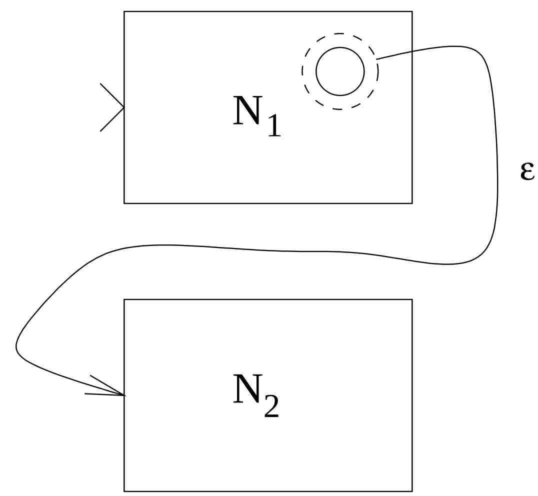
Union

$$L(N) = L(N_1) + L(N_2)$$



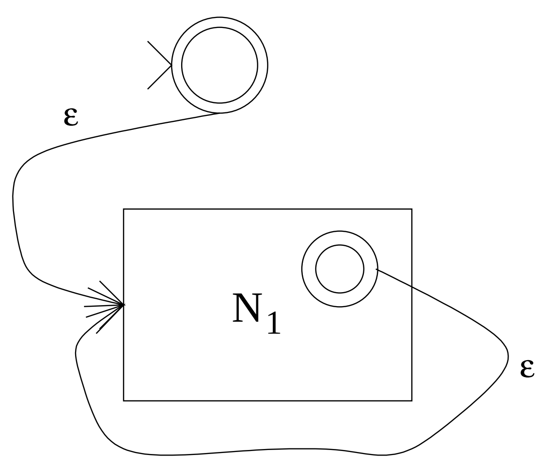
Concatenation

$$L(N) = L(N_1)\, L(N_2)$$



Kleene Star

$$L(N) = (L(N_1))^*$$



$3 \to 4$: Let $N = (\{1, \ldots, n\}, \Sigma, \Delta, 1, F)$, $F = \{f_1, \ldots, f_r\}$

$$L^k_{ij} \;\equiv\; \{w \mid j \in \Delta^\star(i, w);\ \text{no intermediate state} \# > k\}$$

$$\begin{aligned} L^0_{ij} &= \{a \mid j \in \Delta(i, a)\} \;\cup\; \{\epsilon \mid i = j\} \\ L^{k+1}_{ij} &= L^k_{ij} \;\cup\; L^k_{i\,k+1}(L^k_{k+1\,k+1})^\star L^k_{k+1,j} \end{aligned}$$

$$\begin{aligned} e &= L^n_{1\,f_1} \cup \cdots \cup L^n_{1\,f_r} \\ \mathcal{L}(e) &= \mathcal{L}(N) \end{aligned}$$