Reasoning About Collections
Through Efficient Arithmetization

Decision Procedure for Multisets
~presented by Ruzica Piskac~
Verification of Data Structures

```scala
def removeDuplicates(l: List[String]): List[String] = {
  var c = l
  var duplExist = false
  var s: List[String] = List()
  //: (duplExist ↔ S ∪ C ≠ L) ∧ S ∪ C ⊆ L
  while (!c.isEmpty) {
    val elem = c.first
    if (!s.contains(elem)) s = elem :: s
    else duplExist = true
    c = c.tail
  }
  s
  //: duplExist → |S| < |L|
}
```

abstracting lists as multisets
- S denotes s
- C denotes c
- L denotes l
Verification of Data Structures

```scala
def removeDuplicates(l: List[String]): List[String] = {
  var c = l
  var duplExist = false
  var s: List[String] = List()
  //: (duplExist ↔ S ∪ C ≠ L) ∧ S ∪ C ⊆ L
  while (!c.isEmpty) {
    val elem = c.first
    if (!s.contains(elem)) s = elem :: s
    else duplExist = true
    c = c.tail
  }
  s
  //: duplExist → |S| < |L|
}
```

abstracting lists as multisets
- S denotes s
- C denotes c
- L denotes l

\[ S \cup C \neq L \land S \cup C \subseteq L \rightarrow |S| < |L| \]
Reasoning about Multisets

Multiset (bag) is a collection of elements where an element can occur several times.

Formally, multisets is a function $m : E \rightarrow \{0, 1, 2, \ldots \}$
($E$ - finite universe of unknown size)

Operations and Relations on Multisets:

- **Plus:** $(m_1 \cup m_2)(e) = m_1(e) + m_2(e)$
- **Subset:** $m_1 \subseteq m_2 \iff \forall e. m_1(e) \leq m_2(e)$
- **$\forall e. F(m_1(e), \ldots, m_k(e))$, $F$ - QF linear integer arithmetic formula**
- **Cardinality:** $|m| = \sum_{e \in E} m(e)$
Sum Normal Form

\[ P \land (u_1, \ldots, u_n) = \sum_{e \in E} (t_1, \ldots, t_n) \land \forall e. F \]

Every multiset formula can be reduced to its sum normal form.

How can we automatically reason about such sums?
Introducing: LIA*

For a given set of integer vectors $S$,

$$S^* = \{x_1 + \ldots + x_n \mid x_i \in S \land n \geq 0\}$$

**LIA∗ Formulas**

$$P \land (u_1, \ldots, u_n) \in \{(t_1, \ldots, t_n) \mid F; x_1, \ldots, x_p \in \mathbb{N}\}^*$$

where $P$, $F$ are quantifier-free linear integer arithmetic formulas
From Multisets to LIA*

A formula in the sum normal form:

\[ P \land (u_1, \ldots, u_n) = \sum_{e \in E} (t_1, \ldots, t_n) \land \forall e. F \]

is equisatisfiable with the formula

\[ P \land (u_1, \ldots, u_n) \in \{(t'_1, \ldots, t'_n) \mid F; x_1, \ldots, x_p \in \mathbb{N}\}^* \]

Example (continued):

\[(k_0, k_1, k_2) = \sum_{e \in E} (m_o(e), Y(e), X(e)) \land \forall e. m_o(e) = Y(e) + X(e) \]

\[(k_0, k_1, k_2) \in \{(m_o, y, x) \mid m_o = y + x; m_o, y, x \in \mathbb{N}\}^* \]
Semilinear Sets

Let $C_1$ and $C_2$ be sets of vectors of non-negative integers.

$$C_1 + C_2 = \{x_1 + x_2 \mid x_1 \in C_1 \land x_2 \in C_2\}$$

Linear set = set of form $\{x\} + C^*$ for $x \in \mathbb{N}^n$ and $C \subseteq \mathbb{N}^n$ finite

Semilinear set = finite union of linear sets

$$\{2\} + \{10\}^* = \{2, 12, 22, \ldots\} = \{x \mid x = 2 + 10n, n \in \mathbb{N}\}$$

$$\{(3, 3)\} + \{(0, 1), (1, 1)\}^* = \{(x, y) \mid 3 \leq x \land x \leq y\}$$
From LIA* to LIA

- [Ginsburg, Spanier 1968] showed:
  - a solution of a LIA formula is a semilinear set
- Observation:
  - if S is semilinear, then S* can be described in LIA
  - thus, S* is also semilinear
- Consequently:
  \[(u_1, \ldots, u_n) \in \{(t_1, \ldots, t_n) \mid F\}^*\]
  is equivalent to a LIA formula
Complete Reduction Process

- Starting formula: $|X \cup Y| \neq |X| + |Y|$
- LIA* translation:
  
  $k_0 \neq k_1 + k_2 \land (k_0, k_1, k_2) \in \{(m_0, y, x) \mid m_0 = y + x; m_0, y, x \in \mathbb{N}\}^* \subseteq S^*$

- Semilinear set describing $S$:
  
  $S = (0, 0, 0) + \{(1, 0, 1), (1, 1, 0)\}^*$
  
  $(k_0, k_1, k_2) \in S^*$ becomes: $(k_0, k_1, k_2) = \lambda(1, 0, 1) + \mu(1, 1, 0)$

- Final result:
  
  $k_0 \neq k_1 + k_2 \land k_0 = \lambda + \mu \land k_1 = \lambda \land k_2 = \mu$
Summary of Decision Procedure

**Summary:**
- reduce multiset formula to a form $P \land u \in \{z \mid F\}^*$
- find semilinear set $\bigcup_{i=1}^n (a_i + \{b_{i1}, \ldots, b_{ini}\}^*)$ for $F$
- use $a_i, b_{ij}$ to construct formula $F_1$ describing $u \in \{z \mid F\}^*$:

$$\exists \mu_i, \lambda_{ij}. \ (u_1, \ldots, u_n) = \sum_{i=1}^n (\mu_i a_i + \sum_{j=1}^n \lambda_{ij} b_{ij}) \land$$

$$\land_i (\mu_i = 0 \implies \sum_{j=1}^n \lambda_{ij} = o)$$

- check satisfiability of formula $P \land F_1$
Complexity too High?

Problems:
• there can be exponentially many generators of semilinear sets $\rightarrow$ NEXPTIME decision procedure
• computing semilinear sets is hard

Solution:
• if $u$ is generated by $a_i, b_{ij}$, then is generated by polynomial subset of them
Caratheodory Theorem For Integer Cones

Theorem (Eisenbrand, Shmonin)

Let $X \subseteq \mathbb{Z}^d$ be a finite set of integer vectors and let $b \in X^*$. Then there exists $Y \subseteq X$ such that $b \in Y^*$ and $|Y| \leq 2d \log(4dM)$ where $M = \max_{x \in X} \|x\|_\infty$. 

13
Caratheodory Theorem For Integer Cones

Theorem (Eisenbrand, Shmonin)
Let \( X \subseteq \mathbb{Z}^d \) be a finite set of integer vectors and let \( b \in X^* \). Then there exists \( Y \subseteq X \) such that \( b \in Y^* \) and 
\[
|Y| \leq 2d \log(4dM) \quad \text{where} \quad M = \max_{x \in X} \|x\|_\infty.
\]
Caratheodory Theorem For Integer Cones

Theorem (Eisenbrand, Shmonin)

Let $X \subseteq \mathbb{Z}^d$ be a finite set of integer vectors and let $b \in X^*$. Then there exists $Y \subseteq X$ such that $b \in Y^*$ and $|Y| \leq 2d \log(4dM)$ where $M = \max_{x \in X} ||x||_{\infty}$. 
Small Number of Generators

1. \( u = a + b \)
2. apply Eisenbrand-Shmonin theorem as black box on \( b_{ij} \) vectors
3. there are only polynomially vectors \( b_{ij} \) needed to represent \( b \)
4. join them with associated \( a_i \) vectors
5. apply Eisenbrand-Shmonin theorem on remaining \( a_i \) vectors
How to Guess $a_i$ and $b_{ij}$ Vectors?

- Observation: instead of guessing $a_i$, $b_{ij}$, guess solutions of formula $F$
- Result:

$$P \land u \in \{x \mid F(x)\}^*$$

is equisatisfiable with

$$P \land u = \sum_{i \in \{1, \ldots, Q\}} \lambda_i v_i \land \bigwedge_{i \in \{1, \ldots, Q\}} F(v_i)$$

where $Q \in \mathcal{O}(n^2 \log n)$ for formula of a size $n$
Number of Needed Solutions

- In formula

\[ P \land u = \sum_{i \in \{1, \ldots, Q\}} \lambda_i v_i \land \land_{i \in \{1, \ldots, Q\}} F(v_i) \]

*Q is a number (not a variable!)* It depends on
  - dimension of a problem
  - \( \| \|_\infty \) of generating vectors of semilinear sets
  - their size can be computed without actually computing vectors
  - Pottier determined the upped bounds on generators of semilinear sets [Pottier 1991]
Last Hurdle

- Formula

\[ P \land u = \sum_{i \in \{1, \ldots, Q\}} \lambda_i v_i \land \bigwedge_{i \in \{1, \ldots, Q\}} F(v_i) \]

is polynomially large formula:

- but it multiplies variables \( \lambda_i, v_i \) - not linear?
- solution vectors are bounded [Papadimitriou 1981]
- multiplication can be expanded using the ite construct

- Result: \textbf{NP completeness}
Implementation - MUNCH

- the first automated theorem prover for sets and multisets
- based on the computation of semilinear sets
- incomplete, however works for the problems derived in a verification process
- implemented in Scala