<u>Def'n</u> $A \subseteq \mathbb{N}$ is <u>eventually periodic</u> if there are integers M, p s.t. $\forall n > M$ $(n \in A \Leftrightarrow n+p \in A)$

e.g. any finite set, or the non-negative even integers or a finite modification of the evens.

Another corollary of quant elim'n in T:

<u>Corollary</u> $A \subseteq \mathbb{N}$ is definable in $(\mathbb{N}, <, S, +, 0)$ iff A is eventually periodic.

<u>Pf</u>: $(\Rightarrow)$ if A is definable, say by a fmla $\varphi(x)$, then in the extended language w/ $\equiv_m$'s, $\varphi(x)$ is equiv to a q.f. $\varphi^*(x)$ which is an $\wedge, \vee$ combo of $n = x$, $n \neq x$, $n < x$, $n \not< x$, $n \equiv_m x$, $n \not\equiv_m x$

(note: $\varphi(x)$ defines same set in $(\mathbb{N}, <, S, +, 0, \equiv_2, \equiv_3, \ldots)$ as in $(\mathbb{N}, <, S, +, 0)$ so legit to consider just what q.f. version defines)

— convince yourself: each atomic sentence defines eventually periodic set, and any $\wedge\vee$ combo does as well – hence φ^* and hence φ defines ev. per. set, i.e. A is ev. per.

($\Leftarrow$) if $A = \{a_0, \ldots, a_n, a_{n+p}, a_{n+2p}, \ldots\}$ is eventually periodic then A defined in $(\mathbb{N}, <, +, S, 0)$ by the formula

$$x = a_0 \vee \cdots \vee x = a_n \vee \exists m \, (x = a_n + \underbrace{m + \cdots + m}_{p\text{-times}})$$

It follows immediately:

Corollary Multiplication is not definable in $(\mathbb{N}, <, S, +, 0)$

Pf: – if $\cdot$ could be defined then could define any set definable in $(\mathbb{N}, <, S, +, \cdot, 0)$ – but every recursive (in fact: arithmetic) set is definable in $(\mathbb{N}, <, S, +, \cdot, 0)$, and there are many non-eventually-periodic recursive sets.

When Valid_L is decidable

- At beginning of term we proved: if L is lang. containing at least one binary rel'n symbol, or binary function symbol, or two unary function symbols then
$$\text{Valid}_L = \{\sigma \mid \sigma \text{ a valid } L\text{-sentence}\}$$
is not recursive

- Here's a partial converse

Theorem Sps L is a finite lang. containing only unary rel'n symbols $\{P_1, \dots, P_k\}$ and constant symbols $\{c_1, \dots, c_n\}$.
Then Valid_L is recursive.

Pf. - wlog may assume L has no constants:
any sentence $\varphi(c_1, \dots, c_n)$ is valid iff $\forall x_1 \dots \forall x_n \, \varphi(x_1, \dots, x_n)$ is too
(see this by looking @ deduction axioms, or semantically: can interpret the c_i in an arbitrary universe arbitrarily, and any valid φ must be true in all such models.)

- Given a subset $I \subseteq \{1, 2, \dots, k\}$ and $n \geq 1$, define a sentence $\sigma_{I,n}$:
$\exists x_1 \cdots \exists x_n$ ("the x_i are pairwise distinct" $\wedge$
$\bigwedge_{j \leq n} (\bigwedge_{i \in I} P_i(x_j) \wedge \bigwedge_{i \notin I} \neg P_i(x_j)))$

→ How to think about it:

- for a given x in a structure in our lang L, x satisfies some of the P_i and not others
  - i.e. there is $I \subseteq \{1, 2, \dots, k\}$ s.t. $\bigwedge_{i \in I} P_i(x) \wedge \bigwedge_{i \notin I} \neg P_i(x)$
  - Let $\chi_I(x)$ denote this fmla
  - $\sigma_{I,n}$ asserts there are at least n elements satisfying $\chi_I(x)$

- Also note: every x satisfies precisely one of the χ_I
  - so a structure A is specified by partitioning its universe A into 2^k-many sets, one for each $I \subseteq \{1, 2, \dots, k\}$

<u>We claim</u>: each L-formula is equiv to a Boolean combo $(\wedge, \vee, \neg)$ of atomic fmlas and $\sigma_{I,n}$'s

(not quite quantifier elim'n – but close)

Pf: – argue directly by induction on construction of fmlas
– atomic fmlas are atomic, and any boolean combo of a combo of atomics and $\sigma_{I,n}$'s is still a combo of such.
– So sufficient to check for $\exists x \varphi$ where φ a Bool'n combo of atomics, $\sigma_{I,n}$'s
– as usual (by passing to dnf, distrib'ng $\exists x$, going disjunct by disjunct) may assume φ is a conjunction of atomics, $\sigma_{I,n}$'s and negations thereof
– Since $\sigma_{I,n}$'s and $\neg\sigma_{I,n}$'s are sentences, their meaning not changed by $\exists x$, can assume they don't appear (would leave them alone anyway)

– so fmla looks like:
$\exists x(\varphi_1 \wedge \dots \wedge \varphi_n)$
where each φ_i is atomic/neg'n (i.e. fmla is simple!)

– may assume each φ_i involves variable x (o/w leave alone)

- Further may assume no φ_i of form $x = y$ – if so can replace other instances of x w/ y and delete $\exists x$, get qf equiv. fmla.
- So fmla looks like:

$$\exists x \, (x \neq y_1 \wedge \dots \wedge x \neq y_n \wedge P_{i_1}(x) \wedge \dots \wedge P_{i_\ell}(x) \wedge \neg P_{j_1}(x) \wedge \dots \wedge \neg P_{j_m}(x))$$

- may assume $\{i_1, \dots, i_\ell\} \cap \{j_1, \dots, j_m\} = \emptyset$ ow fmla is equiv to $\perp$
- further that $\{i_1, \dots, i_\ell\}$, $\{j_1, \dots, j_m\}$ partition $\{1, \dots, k\}$
  ↳ if not, add all possible cases by disjunction, distrib $\exists x$ again, go disjunct by disjunct
  (e.g. $P_1(x) \wedge \neg P_2(x)$ equiv to $(P_1(x) \wedge \neg P_2(x) \wedge P_3(x)) \vee (P_1(x) \wedge \neg P_2(x) \wedge \neg P_3(x))$)

So:
- using our notation from before fmla is $\exists x (x \neq y_1 \wedge \dots \wedge x \neq y_k \wedge \chi_I(x))$ for $I = \{i_1, \dots, i_\ell\}$

- "there is an x, distinct from each y_i, belonging to precisely the relations P_i, for $i \in I$"

- Can get rid of x by equivalently asserting "some number m of the y_i satisfy $\chi_I(y_i)$, and there are at least $m+1$ elements satisfying χ_I"

- i.e. for each $m \leq n$, let $\chi_{I,m}^{(y_1,\ldots,y_n)}$ assert "there are exactly m (pairwise distinct) elements among the y_i s.t. χ_I"

$$:= \bigvee_{\substack{S \subseteq \{1,\ldots,n\} \\ |S| = m}} \left(\bigwedge_{\substack{i,j \in S \\ (\neq)}} y_i \neq y_j \wedge \bigwedge_{i \in S} \chi_I(y_i) \wedge \bigwedge_{j \notin S} \left(\bigwedge_{i \in S} y_j \neq y_i \Rightarrow \neg \chi_I(y_j) \right) \right)$$

- then our formula equiv to:

$$(\chi_{I,0} \wedge \sigma_{I,1}) \vee (\chi_{I,1} \wedge \sigma_{I,2}) \vee \cdots \vee (\chi_{I,n} \wedge \sigma_{I,m+1})$$

- by inspection: a boolean combo of atomics, $\sigma_{I,k}$'s ✓

- With claim, can prove theorem
- sketch: by claim, any sentence in L equiv to Boolean combo of the $\sigma_{I,n}$'s ← assert $\exists$ at least n elements satisfying $\chi_I(x)$

- neg'n asserts fewer than n such x
- so a Boolean combo just a combo of assertions about the sizes of the sets $S_I = \{x : \chi_I(x)\}$

- can just exhaustively (but algorithmically) check if such a combo is contradictory (e.g. " χ_I has size at most 5 and at least 10 ") or valid (" χ_I has size at most 5 or at least 5 ") or neither (" ... at least 5, at most 10 ")

- Hence $Valid_L$ recursive ✓

We can also get decidability results by restricting class of valid fmlas we want to decide about.

<u>Theorem</u> Sps L is a finite language. Then the class $Valid_L(\forall_1)$ of valid $\forall_1$ fmlas is decidable.

We'll need

<u>Lemma</u> Let Φ be a recursive set of sentences s.t. for each $\sigma \in \Phi$, either σ is valid or it fails in a finite structure. Then checking which fmlas in Φ are valid is ~~[illegible]~~ decidable

<u>Pf.</u> given $\sigma \in \Phi$, recursively enumerate the valid sentences (and check if any are σ) while simultaneously enumerating all finite structures (and checking if σ fails in any). This process halts ✓

<u>Pf of theorem</u>

- we claim that the $\forall_1$ sentences satisfy the ~~[illegible]~~ hypotheses of the Lemma.
- if $\forall x_1 \ldots \forall x_n\, \varphi(x_1, \ldots, x_n)$ is such a sentence, then if not valid, there is a structure A and $a_1, \ldots, a_n \in A$ s.t.

$$A \models \neg \varphi(a_1, \ldots, a_n)$$

- but then can define a ~~[illegible]~~ finite structure $\mathcal{A}'$ w/ universe A' obtained from $\{a_1, \ldots, a_n\}$ by applying $\mathcal{A}$'s functions enough times to check $\neg\varphi(a_1, \ldots, a_n)$
  (possible since φ is q.f.)

- in $\mathcal{A}'$, define all functions ~~[illegible]~~ on inputs whose outputs in A' as in A on other elements arbitrarily.

- relations, constants as in $\mathcal{A}$

- then $\mathcal{A}' \models \neg\varphi(a_1, \ldots, a_n)$

- i.e. $\mathcal{A}'$ is a finite structure in which φ fails ✓