Not all problems can be notived by computers.

- There are uncountably many problems.
- Computers can solve only countably many problems.

An alphabet is a finite set of symbols. $\{0,1\},\{0,1,2,-,9,t,-\}$ ξα, b, ---, Z, A, B, ---, Z}

ASCII

 \geq , \uparrow , \triangle A string over \(\sum is a finite ordered sequence of symbols from 5. {O, 1} 010110 000111

ΣX = the set of all strings over Σ

Proposition: Et is countable. $\frac{P_{ros}f}{\sum} = \bigcup \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{j=1}^{$ えとの $|\Sigma| = |\Sigma| \rightarrow finite$ -> countable.

IX is a countable union of countable sets.

Proposition: $b(\Sigma^*)$ is uncountable.

Language: Any subset of [X.

ASCII alphabet English language in the set of valid English sentences.

Language of integers over $\{0,1,...,9,+,-\}$ $\{0,1,-1,2,-2,3,-3,...\}$ Language of primes $\{2,3,5,7,11,13,...\}$ Problem: Given a language Lover E, and a string of E \(\geq \times, decide whether $\alpha \in L$. English: Checking correctuess. Language of integers: check whether & iz

syntactically valid t 235 2t35 Language of primes: Trimality-testing problem How to represent a language?

L finite $L = \{\alpha_{()}\alpha_{2}, ..., \alpha_{n}\}$ L infinite \rightarrow exhaustive enumeration not possible

Grammars / English-language / Mathematical

English grammar

{ a E M | a is a prime }

C language — grammar

A grammar / description is a ntving over some representation alphabet T, Crammars / Lescriptions E T Only countably many languages can have finite descriptions. Lis specified by such a finite description. Problems that computers can solve C Languages that have finite descriptions

An un solvable problem

HP (The Halfing Problem)

Input: A C program P

An input I for P.

Output: H if Phalts on input I L if P does not halt on injent I

It is impossible to write a Cprogram that nolver (all instances) of the HP.

A diagonalization proof P - a C program E ASCII I - a string E ASCII Each char in ASCII is a sequence of eight bits. BAh Pand I are leinary strings Prijection $\in \rightarrow 1$ $\{0,1\}^* \rightarrow \mathbb{N}$ 001 (1001) = 9

Every input is a natural number.

A natural number can be viewed an a C program or an input.

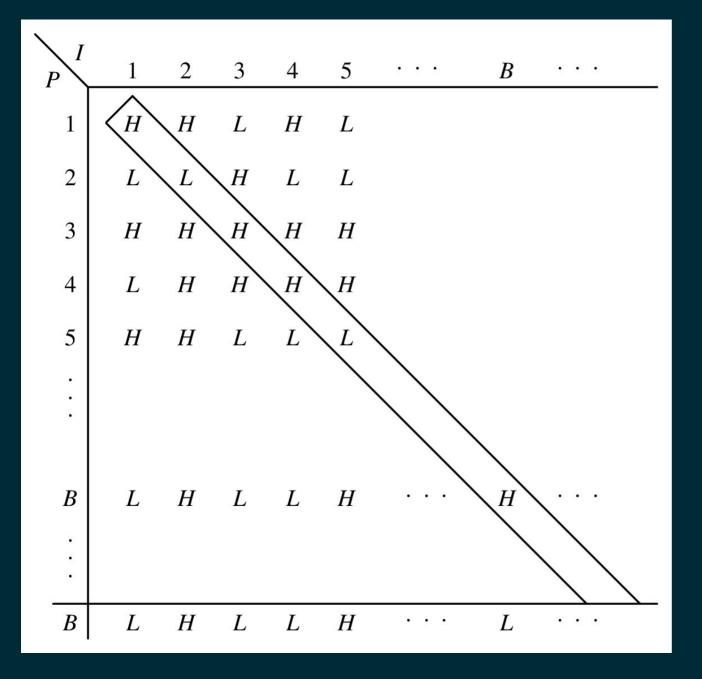
An invalid program can be identified as a program;

main()
{
 exi+(0);

Fa C program A that for all instances of P and I, outputs H or L (as the case is).

Then I can generate a program B! that runs Pon P.

B introduces a contradiction.



Boninfont P: Bruns A on P, P. If A outputs H, B enters an infinite IF A outputs L, B exits. halts on B and B 100ps on B. C