# 木変換の逆正規性保存

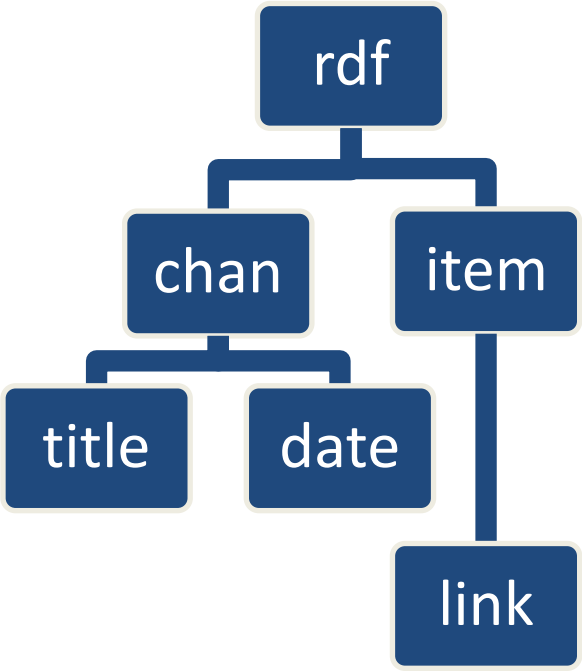
*On Inverse Regularity Preservation*

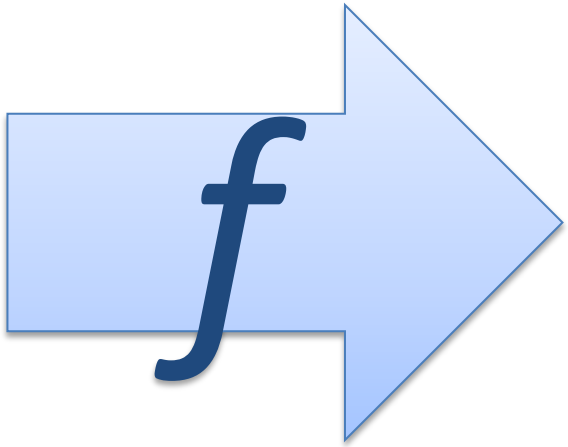
稲葉 一浩（国立情報学研究所）

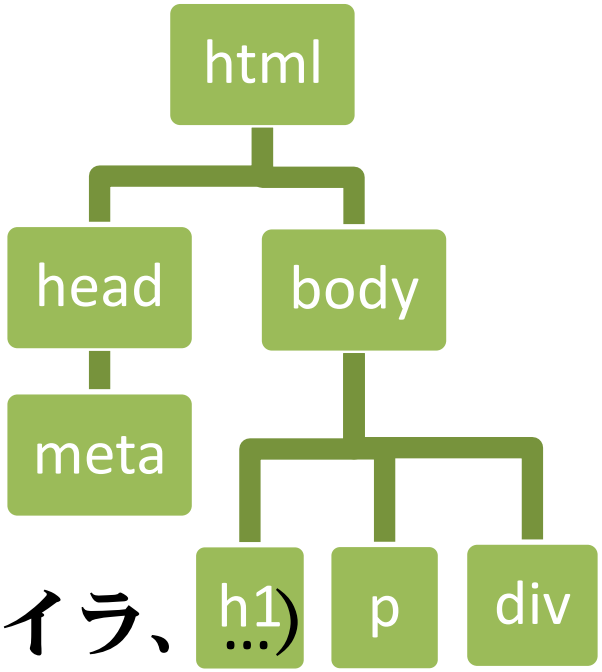
# 木変換とは？

Treeを受け取り

Treeを返す




プログラム

(XML処理、コンパイラ、…)

# 木変換とは？

おことわり

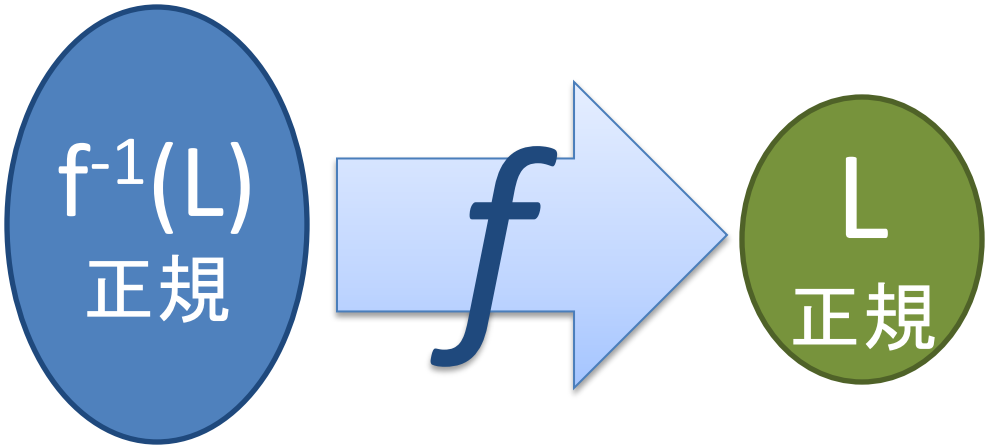
今日は、簡単のため
文字列変換で例示します

# 逆正規性保存とは？

【定義】

f が逆正規性保存 (IRP)　⇔

∀L:正規言語. $f^{-1}(L)=\{t \mid f(t) \in L\}$も正規




例：dup(x) = xx

$dup^{-1}$( a*b*) =　a*|b*

# 発表内容

MTT*という言語のプログラムは
全て逆正規性保存 [EngVog85]

$$\mathbf{MTT^* \subseteq IRP}$$

この包含は proper だろうか？

$\mathbf{MTT^* \subsetneq IRP}$ or $\mathbf{MTT^* = IRP}$ ？

[Today's Talk]

# 背景：IRPの応用例 @ XML

型チェック $f :: L_1 \to L_2$ ？

- XML変換プログラム $f$
- 入力の型(=スキーマ)(= 正規言語) $L_1$
- 出力の型(=スキーマ)(= 正規言語) $L_2$

IRPなら計算可能!

fの型が正しい $\Leftrightarrow f(L_1) \subseteq L_2 \Leftrightarrow L_1 \subseteq f^{-1}(L_2)$

# なぜ逆方向？

- 順正規性保存(FRP) 「∀L:正規. {f(t) | t∈L}も正規」を考えないのは何故か？
  - 理由１：よくある関数はFRPであるよりはIRPである方が多い傾向にある(例えばdup)
  - 理由２：型エラーが起きたときに、IRPで型チェックをする方が良いエラー理由を表示しやすい
    - 「型エラー」＝「$L_1 \subseteq f^{-1}(L_2)$ が成り立たない」＝「$f^{-1}(L_2)$に入らないのに$L_1$に入る反例tがある」＝「その木tを入力したときに、型が合わない結果になってしまう！」という、実際にエラーを起こす**入力例**を提示できる
      - あとはこの入力を使ってステップ実行デバッグするなり出力を計算して眺めてみるなり…
    - FRPでは「型エラー」＝「$f(L_1) \subseteq L_2$ が成り立たない」＝「$L_2$に入らないのに$f(L_1)$に入る反例tがある」＝「なにか下手な入力を入れると、こんなまずい出力がでちゃうよ！」という、エラー時の**出力例**を提示
      - じっと睨んで、こんなのが出てしまう理由を推測する位しかできない

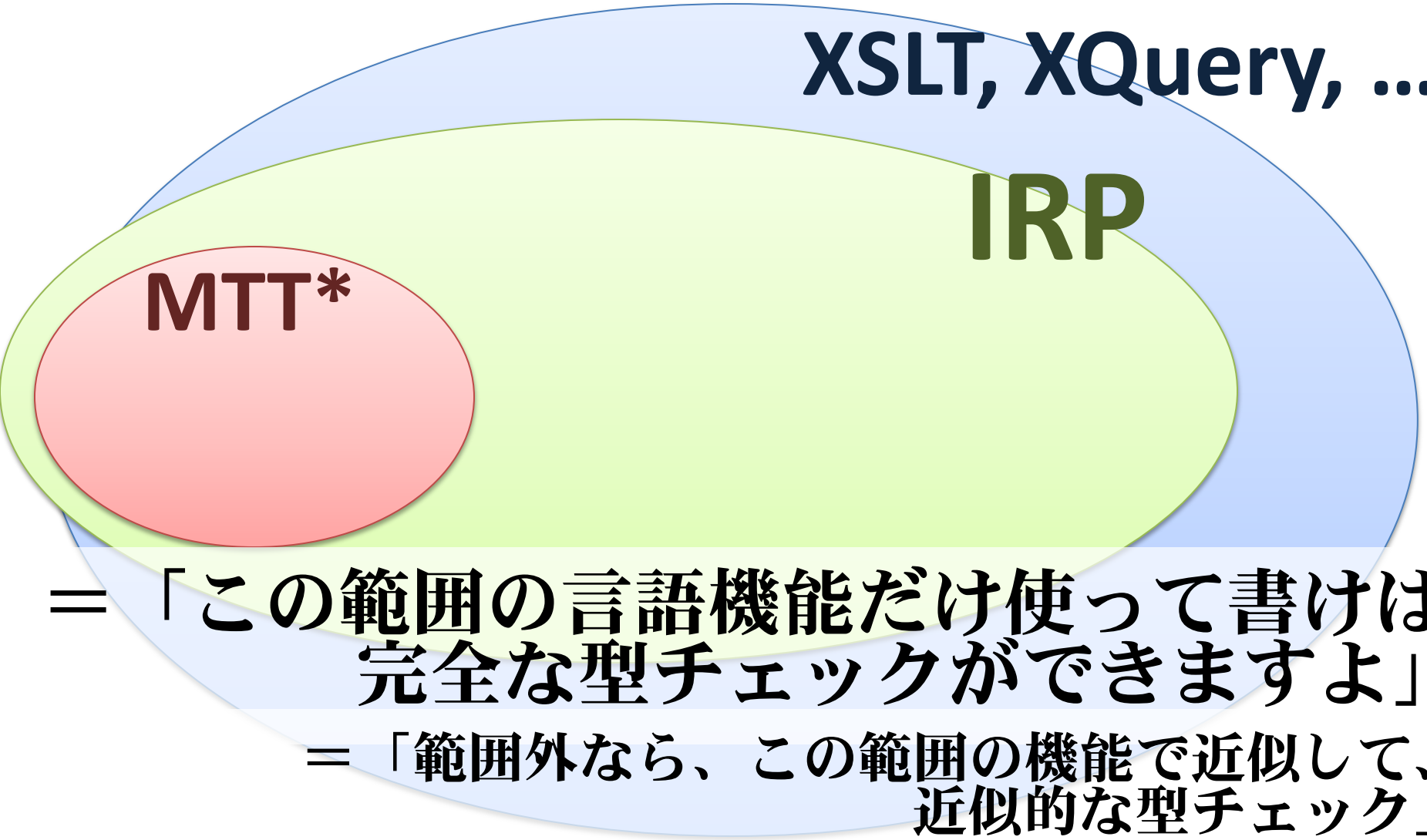
# 現実の(XML処理)言語は...





当然、IRPとは限らない

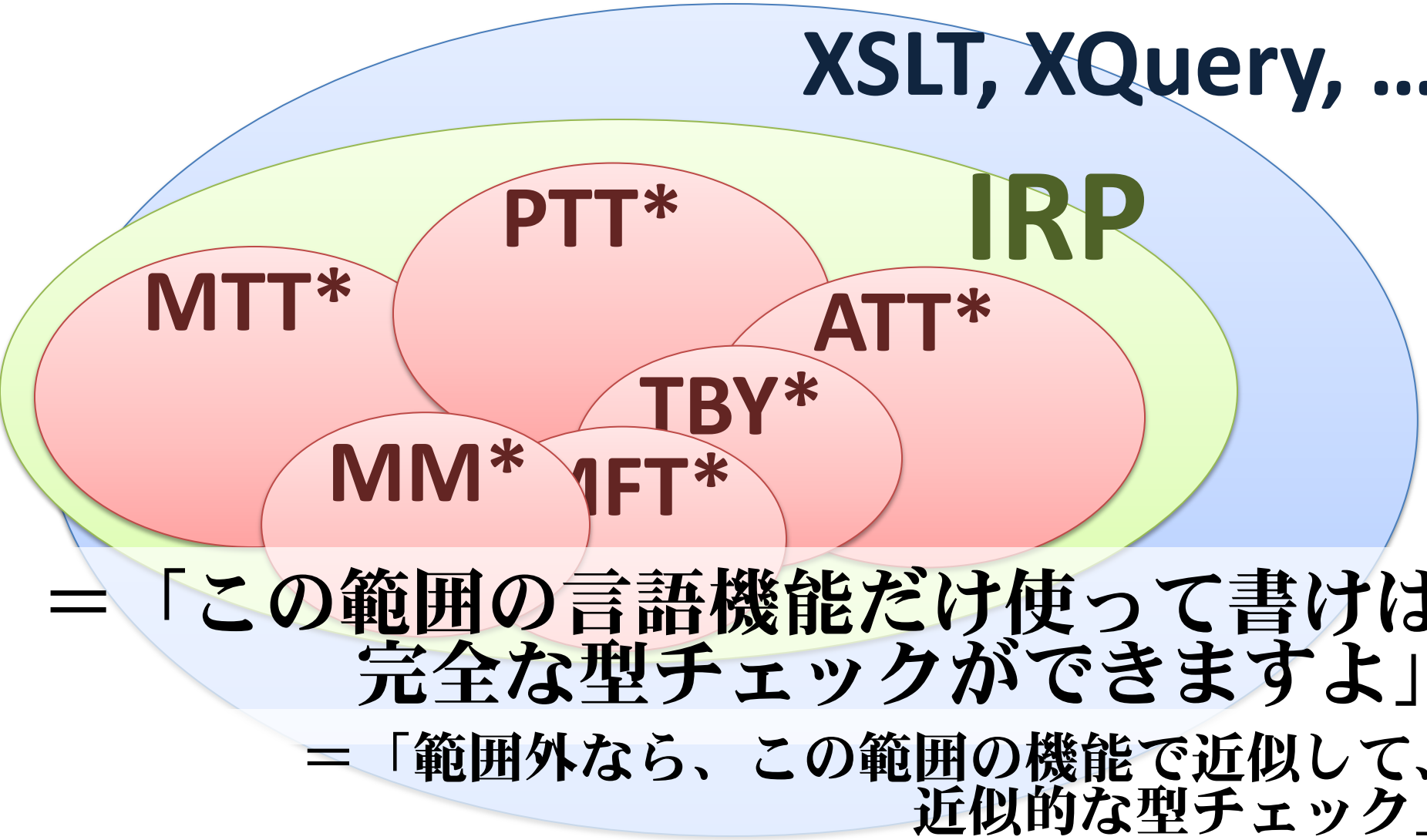
# 制限された言語を考える




＝「この範囲の言語機能だけ使って書けば、
完全な型チェックができますよ」

＝「範囲外なら、この範囲の機能で近似して、
近似的な型チェック」

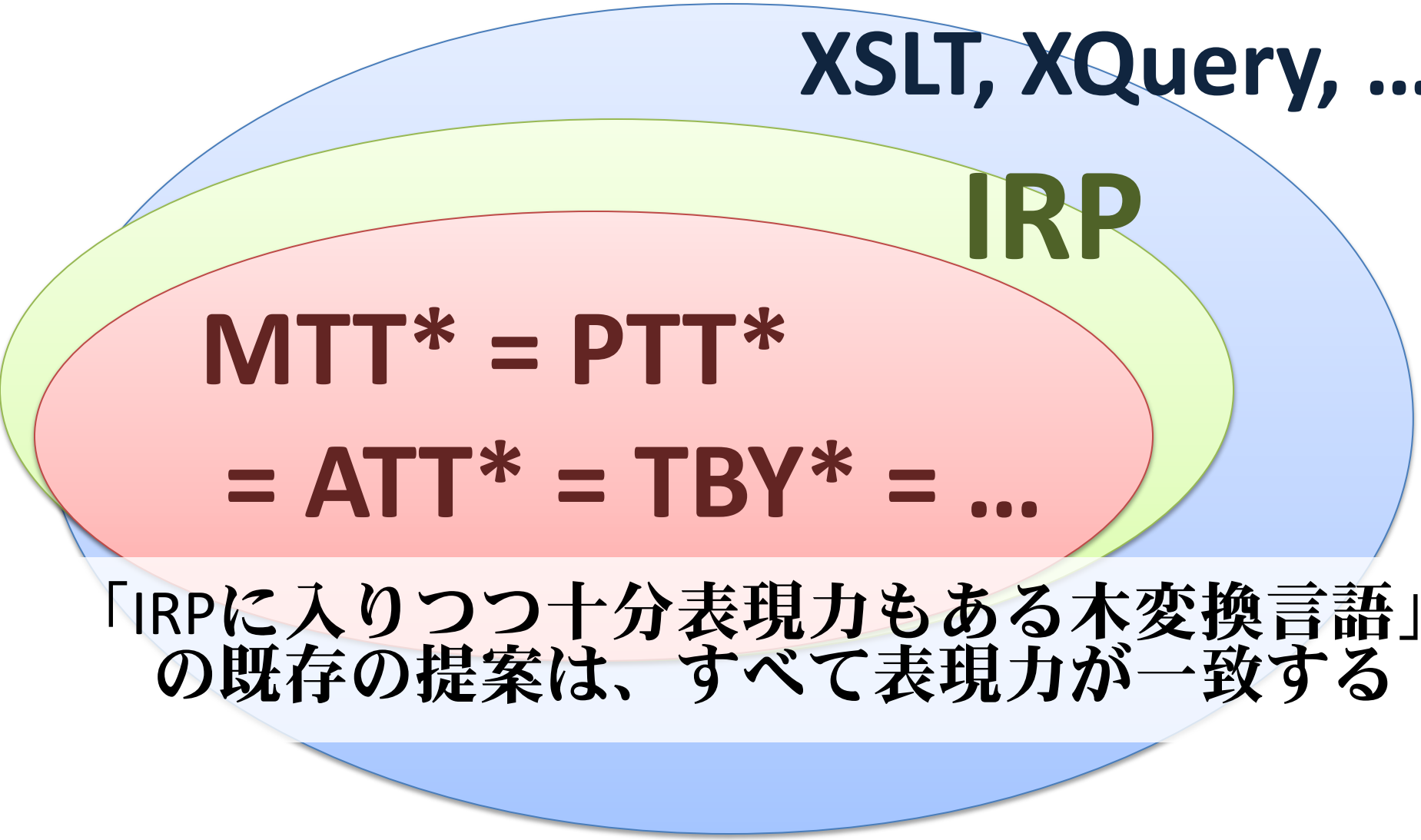
# 制限された言語を考える




＝「この範囲の言語機能だけ使って書けば、
完全な型チェックができますよ」

＝「範囲外なら、この範囲の機能で近似して、
近似的な型チェック」

# それぞれのカバー範囲は？



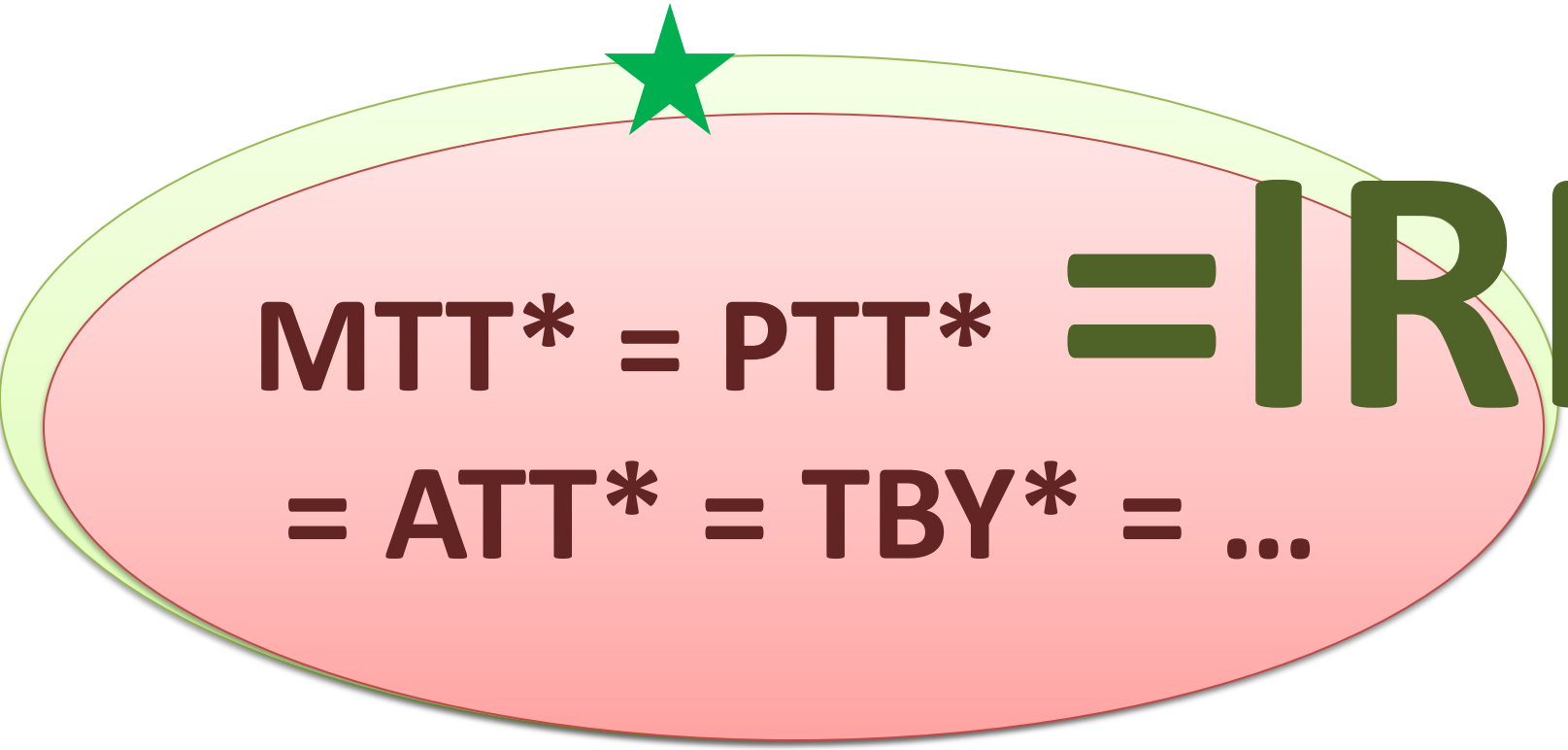

「IRPに入りつつ十分表現力もある木変換言語」
の既存の提案は、すべて表現力が一致する！

# Q: 実はIRPも一致するのでは…？

MTT* = PTT* = ATT* = TBY* = … =IRP?

# Q: 実はIRPも一致するのでは…？

The Answer is **No**!



# 証明 (1)

文字列を超指数的に長くする関数

$$\mathrm{tower}(\underbrace{\text{“a..a”}}_{n}) = \underbrace{\text{“aa...aa”}}_{2\hat{}\hat{}n}$$

where $2\hat{}\hat{}0 = 1$, $2\hat{}\hat{}(n+1) = 2^{2\hat{}\hat{}n}$

は MTT* で表現できない [EngVog85]

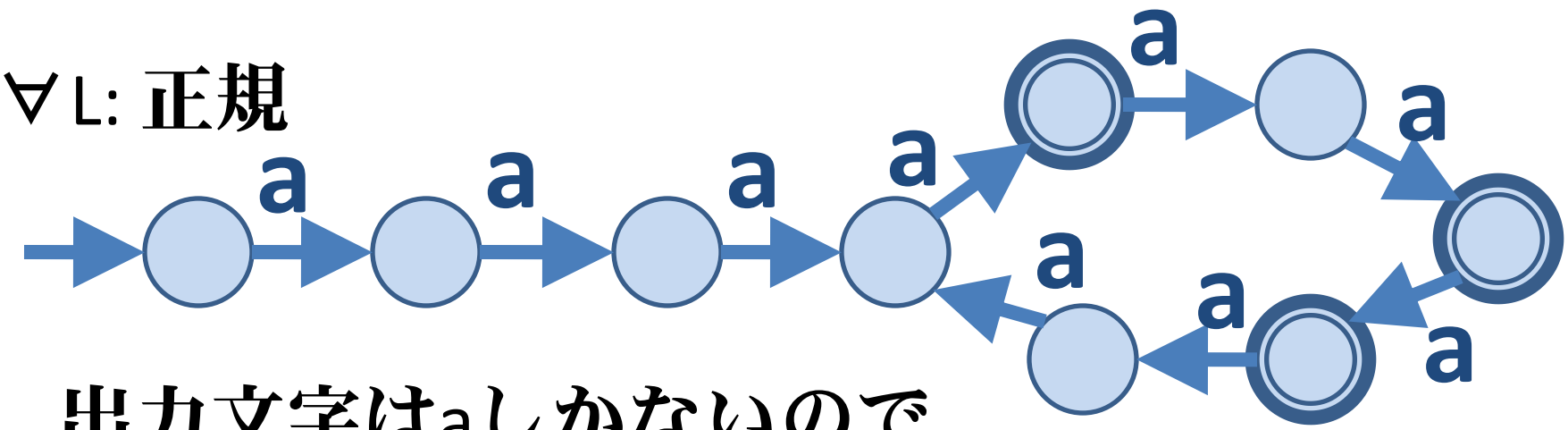
∵「MTT* ≒構造再帰と関数合成のみで書ける木変換」では指数的増加しか書けない

# 証明 (2.1)

## tower 関数は IRP である

つまり　∀L:正規. {x |tower(x)∈L} も正規

∀L: 正規




出力文字はaしかないので

$L = \{ a^n \mid n \equiv r_1 \text{ or } \dots \text{ or } n \equiv r_k \bmod m \}$ の形

(※ただし有限個の違いは除く)

# 証明 (2.2)

$L = \{ a^n \mid n \equiv r_1, \ldots, r_k \bmod m \}$ について

{x | tower{x}∈L}

= $\{a^n \mid 2\text{^^}n \equiv r_1, \ldots, r_n \bmod m\}$ は正規？

Yes! 2^^n mod m は、(※有限個のnを除き) 定数

なのでこの集合は(※有限個を除き)全体集合か空集合

– m が奇素数 → フェルマーの小定理より

2^^n+1 ＝ 2^(2^^n) ≡ 2^(2^^n mod m-1) ＝ 2^定数

– m = 2 → 常に0 なので定数

– m が合成数 → 中国剰余定理より素数に帰着

# まとめ

示したこと

$$\mathrm{MTT}^* \subsetneq \mathrm{IRP}$$

未解決問題：towerより現実的な例はあるか？

$$\mathrm{MTT}^* \cap \mathrm{LSI} \subsetneq \mathrm{IRP} \cap \mathrm{LSI} ?$$

※ LSI = Linear Size Increase

i.e., exists $c > 0$, forall $t$, $\mathrm{size}(f(t)) \leqq c\,\mathrm{size}(t)$

# 参考文献

“Regularity Preserving Functions”

[Kosaraju74] [Seiferas&McNaughton76, Kozen96, Zhang99]

- f : N → N が Regularity Preservingとは
  「∀L:正規言語. {x | ∃y. f(|x|)=|y| & xy∈L}も正規」
- “Inverse” という形容はないが、本質的に
  Alphabet 1 種類の文字列変換の IRP と同値
- RP　iff　f is “ultimately periodic“ modulo any n∈N
- 接尾辞以外の除去への拡張等 [Matos06, **Berstel et al., 06**]
  - **(2010/03/11 追記)**
    **tower が IRP であることも [Berstel et al., 06] に書いてありました！**
    **http://dx.doi.org/10.1016/j.tcs.2005.11.034**

# MTT*とは [Engelfriet&Vogler85]

(for MTT* ∩ LSI, see [Engelfriet&Maneth03])

MTT = 木の上の(相互)構造再帰＋ 累積変数

- 型 Tree(Σ) × Tree(Δ)$^k$ → Tree(Δ) の関数の集まり

MTT* = 有限個のMTTの合成

MTT ::= FUN … FUN
FUN ::= f(A($x_1$,…,$x_n$), $y_1$,…,$y_k$) → RHS
RHS ::= C( RHS, … , RHS )
| f($x_i$, RHS, …, RHS) | $y_i$

例：

start( A($x_1$) ) → double( $x_1$, double($x_1$, E) )
double( A($x_1$), $y_1$) → double( $x_1$, double($x_1$, $y_1$) )
double( B, $y_1$ ) → C( $y_1$, $y_1$ )

# 主補題の証明 (mで帰納法)

$\forall m \exists k_m \forall n \geqq k_m.\ (2\text{^^}k_m \equiv 2\text{^^}n \bmod m)$

- m = 1: $k_m$=0 とすると以降全て $\equiv 0 \bmod m$
- m = 2: $k_m$=1 とすると以降全て $\equiv 0 \bmod m$
- m = p (奇素数):
  - フェルマーの小定理から $2^{p-1} \equiv 1 \bmod p$ より
  - $2\text{^^}n+1 = 2^{2\text{^^}n} \equiv 2^{(2\text{^^}n \,\%\, p-1)} \bmod p$ である
  - 帰納法の仮定より $n \geqq k_{p-1}$ 以上で $2\text{^^}n \,\%\, p-1$ は定数
  - ゆえに $k_p = k_{p-1} + 1$ とすれば題意を満たす
- m = a b (合成数, a,b は互素):
  - 中国剰余定理より、二数が mod a, mod b で等しければ mod ab でも等しいゆえに $k_m = \max(k_a, k_b)$ と取ればよい

# 証明: tower∈IRP

- $L \subseteq a^*$ を正規言語とする
  - Folklore: （決定性オートマトンの構造より）
    - L は $L_f \cup (L_r / L_b)$ という形。ただし
    - $L_f, L_b$ は有限集合
    - $L_r = \{a^n \mid n\%m \in \{r_1,\ldots,r_k\}\}$ for some m & $r_1, \ldots, r_k < m$
  - 逆写像の集合論的性質より
    - $\text{tower}^{-1}(L) = \text{tower}^{-1}(L_f) \cup (\text{tower}^{-1}(L_r) / \text{tower}^{-1}(L_b))$
  - tower は単射なので、有限集合の逆像は有限集合
    - $\text{tower}^{-1}(L_f)$ と $\text{tower}^{-1}(L_b)$ は有限。よって正規言語
  - 主補題より
    - $\text{tower}^{-1}(L_r) = L_1 \cup L_2$
    - $L_1 = \{a^n \mid n < k_m, f(a^n) \in L\}$ 有限なので正規
    - $L_2 = \{a^n \mid n \geqq k_m\}$ if $2\text{^^}k_m\%m \in \{r_1,\ldots,r_k\}$ or $L_2 = \Phi$ いずれにせよ正規
  - 正規言語は ∪ と / で閉じているので、$\text{tower}^{-1}(L)$ も正規

# 証明: tower∉MTT*

- Theorem 3.24 of [EngVog85]
  - $f \in MTT \Rightarrow \exists c.\ \forall s.\ len(f(s)) \leqq c^{len(s)}$
  - 証明はSentential Formと入力sに関する帰納法
- MTT* = MTT の有限合成で書ける変換
  - $tower = f_1 \circ f_2 \circ \ldots \circ f_k \in MTT^*$ (where $f_i \in MTT$) とせよ
  - 先ほどの定理から
    - $\exists c_1, c_2, \ldots, c_k . \forall s . len(tower(s)) \leqq c_k \hat{}(c_{k-1} \hat{} \ldots (c_1 \hat{} len(s)) \ldots)$
- ^^ は任意の指数関数より速く増加する
  - 証明は以下をkに関する帰納法で
    $\forall k \geqq 1 \forall q>0\ \forall r \exists m \in Nat\ \forall\ n \geqq m .$
    $q \cdot 2\hat{}\hat{}n \geqq c_k \hat{}(c_{k-1} \hat{} \ldots (c_1 \hat{} n) \ldots)$
- ゆえに矛盾