There exists a non-recursively enumerable set $\{n \in \mathbb{N} : \varphi(n)\}$ which is co-recursively enumerable, where the formula $\varphi(n)$ is short and can be easily translated into a first-order formula in Peano arithmetic

Apoloniusz Tyszka
Hugo Kołłątaj University
Balicka 116B, 30-149 Kraków, Poland
E-mail: *rttyszka@cyf-kr.edu.pl*

Abstract

We prove that the set

$$W = \{n \in \mathbb{N} :\ \exists p,q \in \mathbb{N}\ ((2n = (p+q)(p+q+1)+2q) \wedge$$

$$\forall (x_0,\dots,x_p) \in \mathbb{N}^{p+1}\ \ \exists (y_0,\dots,y_p) \in \{0,\dots,q\}^{p+1}$$

$$((\forall j,k \in \{0,\dots,p\}\ (x_j+1=x_k \Rightarrow y_j+1=y_k)) \wedge$$

$$(\forall i,j,k \in \{0,\dots,p\}\ (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k)))))\}$$

is not recursively enumerable. We prove that the set $\mathbb{N} \setminus W$ is recursively enumerable. Let $\beta : \mathbb{N}^3 \to \mathbb{N}$ denote Gödel's β function. For $x_1, x_2, x_3 \in \mathbb{N}$, $\beta(x_1,x_2,x_3)$ equals the remainder after integer division of x_1 by $1+(x_3+1)\cdot x_2$. We prove that the set W consists of all $n \in \mathbb{N}$ such that

$$\forall u,v \in \mathbb{N}\ \exists a,b,p,q \in \mathbb{N}\ ((2n = (p+q)(p+q+1)+2q)\ \wedge\ \forall i,j,k \in \{0,\dots,p\}$$

$$((\beta(a,b,i) \leqslant q)\ \wedge\ (\beta(u,v,j)+1=\beta(u,v,k) \Rightarrow \beta(a,b,j)+1=\beta(a,b,k)) \wedge$$

$$(\beta(u,v,i)\cdot\beta(u,v,j)=\beta(u,v,k) \Rightarrow \beta(a,b,i)\cdot\beta(a,b,j)=\beta(a,b,k))))$$

The above formula can be easily translated into a first-order formula in Peano arithmetic.

2020 Mathematics Subject Classification: 03D25.

Key words and phrases: computable function, eventual domination, Gödel's β function, limit-computable function, recursively enumerable set, undecidable decision problem.

Semi-algorithms differ from algorithms, as they may not terminate.

Definition 1. *(cf. [4, pp. 233–235]). A computation in the limit of a function $f : \mathbb{N} \to \mathbb{N}$ is a semi-algorithm which takes as input a non-negative integer n and for every $m \in \mathbb{N}$ prints a non-negative integer $\xi(n,m)$ such that $\lim\limits_{m \to \infty} \xi(n,m) = f(n)$.*

By Definition 1, a function $f : \mathbb{N} \to \mathbb{N}$ is computable in the limit when there exists an infinite computation which takes as input a non-negative integer n and prints a non-negative integer on each iteration and prints $f(n)$ on each sufficiently high iteration.

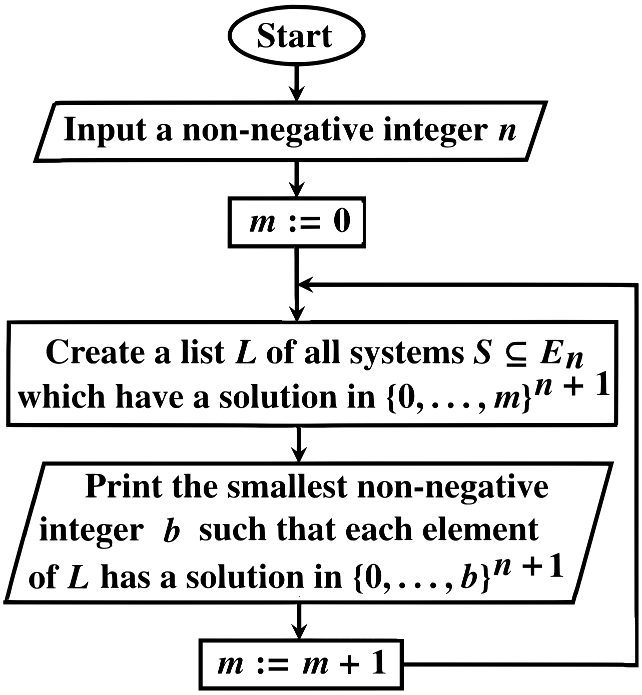
For $n \in \mathbb{N}$, let

$$E_n = \{1 = x_k,\ x_i + x_j = x_k,\ x_i \cdot x_j = x_k :\ i, j, k \in \{0, \ldots, n\}\}$$

Theorem 1. *([3, p. 118]). There exists a limit-computable function $f : \mathbb{N} \to \mathbb{N}$ which eventually dominates every computable function $g : \mathbb{N} \to \mathbb{N}$.*

For $n \in \mathbb{N}$, $f(n)$ denotes the smallest $b \in \mathbb{N}$ such that if a system of equations $S \subseteq E_n$ has a solution in $\mathbb{N}^{n+1}$, then S has a solution in $\{0, \ldots, b\}^{n+1}$. The function $f : \mathbb{N} \to \mathbb{N}$ is computable in the limit and eventually dominates every computable function $g : \mathbb{N} \to \mathbb{N}$, see [6]. The term *"dominated"* in the title of [6] means *"eventually dominated"*.

Theorem 2. *([6]). Flowchart 1 shows a semi-algorithm which computes $f(n)$ in the limit.*



Flowchart 1

A semi-algorithm which computes $f(n)$ in the limit

Definition 2. *An approximation of a tuple $(x_0, \ldots, x_n) \in \mathbb{N}^{n+1}$ is a tuple $(y_0, \ldots, y_n) \in \mathbb{N}^{n+1}$ such that*

$$(\forall k \in \{0, \ldots, n\}\ (1 = x_k \Rightarrow 1 = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, n\}\ (x_i + x_j = x_k \Rightarrow y_i + y_j = y_k)) \wedge$$

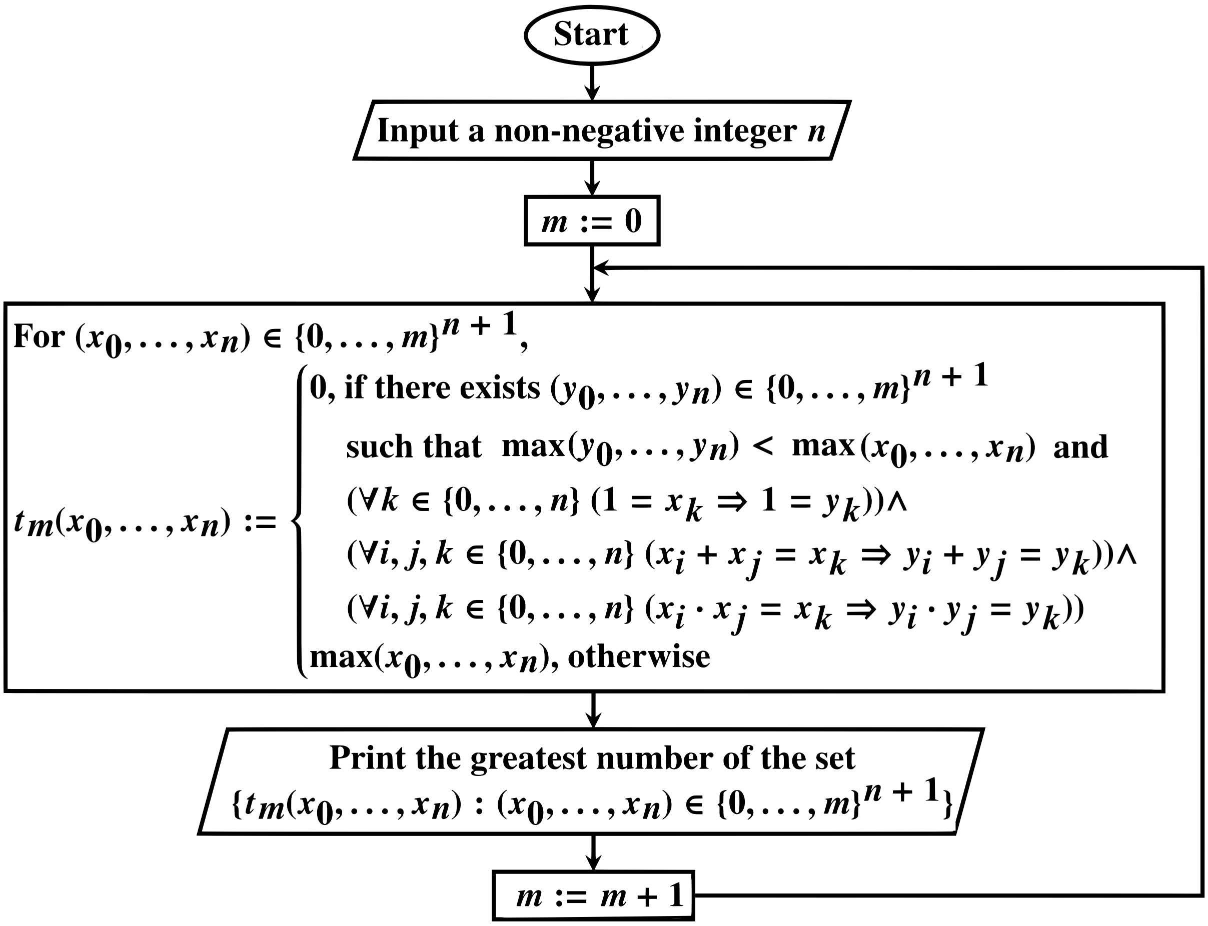
$$(\forall i, j, k \in \{0, \ldots, n\}\ (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k))$$

Observation 1. *For every $n \in \mathbb{N}$, there exists a set $A(n) \subseteq \mathbb{N}^{n+1}$ such that*

$$\mathrm{card}(A(n)) \leqslant 2^{\mathrm{card}(E_n)} = 2^{n+1+2\cdot(n+1)^3}$$

and every tuple $(x_0, \ldots, x_n) \in \mathbb{N}^{n+1}$ possesses an approximation in $A(n)$.

Flowchart 2 shows a simpler semi-algorithm which computes $f(n)$ in the limit.



Flowchart 2

A simpler semi-algorithm which computes $f(n)$ in the limit

Lemma 1. *For every $n, m \in \mathbb{N}$, the number printed by Flowchart 2 does not exceed the number printed by Flowchart 1.*

Proof. For every $(a_0, \dots, a_n) \in \{0, \dots, m\}^{n+1}$,

$$E_n \supseteq \{1 = x_k :\ (k \in \{0, \dots, n\}) \wedge (1 = a_k)\} \cup$$

$$\{x_i + x_j = x_k :\ (i, j, k \in \{0, \dots, n\}) \wedge (a_i + a_j = a_k)\} \cup$$

$$\{x_i \cdot x_j = x_k :\ (i, j, k \in \{0, \dots, n\}) \wedge (a_i \cdot a_j = a_k)\}$$

□

Lemma 2. *For every $n, m \in \mathbb{N}$, the number printed by Flowchart 1 does not exceed the number printed by Flowchart 2.*

Proof. Let $n, m \in \mathbb{N}$. For every system of equations $S \subseteq E_n$, if $(a_0, \dots, a_n) \in \{0, \dots, m\}^{n+1}$ and $(a_0, \dots, a_n)$ solves S, then $(a_0, \dots, a_n)$ solves the following system of equations:

$$\{1 = x_k :\ (k \in \{0, \dots, n\}) \wedge (1 = a_k)\} \cup$$

$$\{x_i + x_j = x_k :\ (i, j, k \in \{0, \dots, n\}) \wedge (a_i + a_j = a_k)\} \cup$$

$$\{x_i \cdot x_j = x_k :\ (i, j, k \in \{0, \dots, n\}) \wedge (a_i \cdot a_j = a_k)\}$$

□

Theorem 3. *For every* $n, m \in \mathbb{N}$*, Flowcharts 1 and 2 print the same number.*

Proof. It follows from Lemmas 1 and 2. ☐

Corollary 1. *For every* $n, m \in \mathbb{N}$*, Flowcharts 1 and 2 print the smallest* $b \in \{0, \ldots, m\}$ *such that every tuple* $(x_0, \ldots, x_n) \in \{0, \ldots, m\}^{n+1}$ *possesses an approximation in* $\{0, \ldots, b\}^{n+1}$*.*

Theorem 4. *For every* $n \in \mathbb{N}$*,* $f(n)$ *is the smallest* $b \in \mathbb{N}$ *such that every tuple* $(x_0, \ldots, x_n) \in \mathbb{N}^{n+1}$ *possesses an approximation in* $\{0, \ldots, b\}^{n+1}$*.*

Proof. It follows from Theorem 2 and Corollary 1. ☐

Theorem 5. *No algorithm takes as input non-negative integers* n *and* m *and decides whether or not*

$$\forall (x_0, \ldots, x_n) \in \mathbb{N}^{n+1} \;\; \exists (y_0, \ldots, y_n) \in \{0, \ldots, m\}^{n+1}$$

$$((\forall k \in \{0, \ldots, n\} \; (1 = x_k \Rightarrow 1 = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, n\} \; (x_i + x_j = x_k \Rightarrow y_i + y_j = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, n\} \; (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k)))$$

Proof. Since the function f is not computable, it follows from Theorem 4. ☐

Lemma 3. *([2]). The function*

$$\mathbb{N}^2 \ni (p, q) \to \frac{1}{2}(p+q)(p+q+1) + q \in \mathbb{N}$$

is bijective.

Theorem 6. *No algorithm takes as input a non-negative integer* n *and decides whether or not*

$$\exists p, q \in \mathbb{N} \; ((2n = (p+q)(p+q+1) + 2q)) \wedge$$

$$\forall (x_0, \ldots, x_p) \in \mathbb{N}^{p+1} \;\; \exists (y_0, \ldots, y_p) \in \{0, \ldots, q\}^{p+1}$$

$$((\forall k \in \{0, \ldots, p\} \; (1 = x_k \Rightarrow 1 = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, p\} \; (x_i + x_j = x_k \Rightarrow y_i + y_j = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, p\} \; (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k))))$$

Proof. It follows from Theorem 5 and Lemma 3. ☐

Let

$$T = \{n \in \mathbb{N} : \; \exists p, q \in \mathbb{N} \; ((2n = (p+q)(p+q+1) + 2q) \wedge$$

$$\forall (x_0, \ldots, x_p) \in \mathbb{N}^{p+1} \;\; \exists (y_0, \ldots, y_p) \in \{0, \ldots, q\}^{p+1}$$

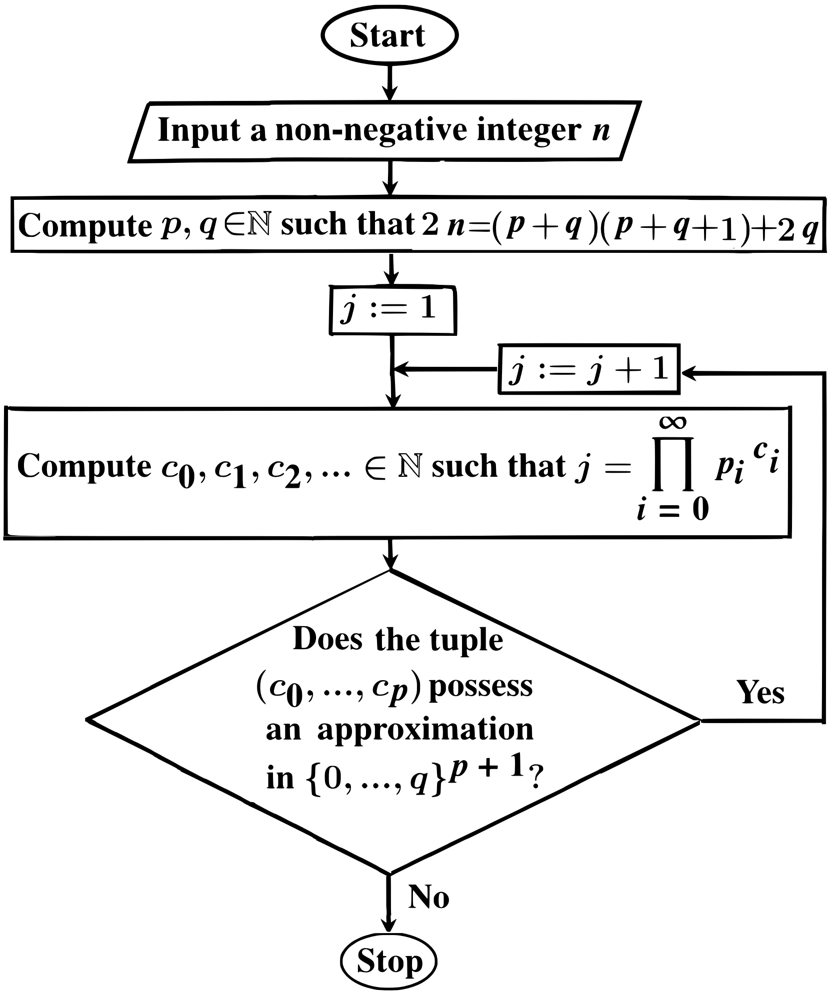
$$((\forall k \in \{0, \ldots, p\} \; (1 = x_k \Rightarrow 1 = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, p\} \; (x_i + x_j = x_k \Rightarrow y_i + y_j = y_k)) \wedge$$

$$(\forall i, j, k \in \{0, \ldots, p\} \; (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k))))\}$$

Theorem 7. *The set* $\mathbb{N} \setminus T$ *is recursively enumerable.*

Proof. For $i \in \mathbb{N}$, let p_i denote the i-th prime number. Flowchart 3 shows a semi-algorithm which takes as input $n \in \mathbb{N}$ and terminates if and only if $n \in \mathbb{N} \setminus T$.



Flowchart 3

A semi-algorithm which takes as input $n \in \mathbb{N}$ and terminates if and only if $n \in \mathbb{N} \setminus T$

□

Theorem 8. *The set* T *is not recursively enumerable.*

Proof. It follows from Theorems 6 and 7. □

Lemma 4. *([5, p. 110]). For non-negative integers, the equation* $x + y = z$ *is equivalent to a system which consists of equations of the forms* $v + 1 = w$ *and* $u \cdot v = w$.

For $n \in \mathbb{N}$, $h(n)$ denotes the smallest $b \in \mathbb{N}$ such that if a system of equations $S \subseteq \{x_j + 1 = x_k,\ x_i \cdot x_j = x_k :\ i, j, k \in \{0, \ldots, n\}\}$ has a solution in $\mathbb{N}^{n+1}$, then S has a solution in $\{0, \ldots, b\}^{n+1}$. From Lemma 4 and [6], it follows that the function $h : \mathbb{N} \to \mathbb{N}$ is computable in the limit and eventually dominates every computable function $g : \mathbb{N} \to \mathbb{N}$.

Theorem 9. *No algorithm takes as input non-negative integers* n *and* m *and decides whether or not*

$$\forall (x_0,\ldots,x_n) \in \mathbb{N}^{n+1}\ \exists (y_0,\ldots,y_n) \in \{0,\ldots,m\}^{n+1}$$
$$((\forall j,k \in \{0,\ldots,n\}\ (x_j+1=x_k \Rightarrow y_j+1=y_k)) \wedge$$
$$(\forall i,j,k \in \{0,\ldots,n\}\ (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k)))$$

Proof. It holds because the function h is not computable, and for every $n \in \mathbb{N}$, $h(n)$ is the smallest $b \in \mathbb{N}$ such that

$$\forall (x_0,\ldots,x_n) \in \mathbb{N}^{n+1}\ \exists (y_0,\ldots,y_n) \in \{0,\ldots,b\}^{n+1}$$
$$((\forall j,k \in \{0,\ldots,n\}\ (x_j+1=x_k \Rightarrow y_j+1=y_k)) \wedge$$
$$(\forall i,j,k \in \{0,\ldots,n\}\ (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k)))$$

□

Theorem 10. *No algorithm takes as input a non-negative integer* n *and decides whether or not*

$$\exists p,q \in \mathbb{N}\ ((2n=(p+q)(p+q+1)+2q)\wedge$$
$$\forall (x_0,\ldots,x_p) \in \mathbb{N}^{p+1}\ \exists (y_0,\ldots,y_p) \in \{0,\ldots,q\}^{p+1}$$
$$((\forall j,k \in \{0,\ldots,p\}\ (x_j+1=x_k \Rightarrow y_j+1=y_k)) \wedge$$
$$(\forall i,j,k \in \{0,\ldots,p\}\ (x_i \cdot x_j = x_k \Rightarrow y_i \cdot y_j = y_k))))$$

Proof. It follows from Theorem 9 and Lemma 3. □

Let W denote the algorithmically undecidable subset of $\mathbb{N}$ considered in Theorem 10. Similarly as in Theorem 7, the set $\mathbb{N} \setminus W$ is recursively enumerable. Similarly as in Theorem 8, the set W is not recursively enumerable. Let $\beta : \mathbb{N}^3 \to \mathbb{N}$ denote Gödel's β function, see [1]. For $x_1,x_2,x_3 \in \mathbb{N}$, $\beta(x_1,x_2,x_3)$ equals the remainder after integer division of x_1 by $1+(x_3+1)\cdot x_2$.

Lemma 5. *([1]). If* $(d_0,\ldots,d_p) \in \mathbb{N}^{p+1}$*, then* $\exists b,c \in \mathbb{N}\ \forall l \in \{0,\ldots,p\}\ \beta(b,c,l)=d_l$.

Theorem 11. *The formula that defines the set* W *can be easily translated into a first-order formula in Peano arithmetic.*

Proof. By Lemma 5, the set W consists of all $n \in \mathbb{N}$ such that

$$\forall u,v \in \mathbb{N}\ \exists a,b,p,q \in \mathbb{N}\ ((2n=(p+q)(p+q+1)+2q)\ \wedge\ \forall i,j,k \in \{0,\ldots,p\}$$
$$((\beta(a,b,i) \leqslant q)\ \wedge\ (\beta(u,v,j)+1=\beta(u,v,k) \Rightarrow \beta(a,b,j)+1=\beta(a,b,k)) \wedge$$
$$(\beta(u,v,i)\cdot\beta(u,v,j)=\beta(u,v,k) \Rightarrow \beta(a,b,i)\cdot\beta(a,b,j)=\beta(a,b,k))))$$

The above formula can be easily translated into a first-order formula in Peano arithmetic. □

References

[1] *Gödel's β function, https://en.wikipedia.org/wiki/G%C3%B6del%27s_%CE%B2_function*.

[2] *Pairing function, https://en.wikipedia.org/wiki/Pairing_function*.

[3] J. S. Royer and J. Case, *Subrecursive Programming Systems: Complexity and Succinctness,* Birkhäuser, Boston, 1994.

[4] R. I. Soare, *Interactive computing and relativized computability,* in: B. J. Copeland, C. J. Posy, and O. Shagrir (eds.), Computability: Turing, Gödel, Church and beyond, MIT Press, Cambridge, MA, 2013, 203–260.

[5] A. Tyszka, *A hypothetical upper bound on the heights of the solutions of a Diophantine equation with a finite number of solutions*, Open Comput. Sci. 8 (2018), no. 1, 109–114, *https://doi.org/10.1515/comp-2018-0012*.

[6] A. Tyszka, *All functions $g:\mathbb{N}\to\mathbb{N}$ which have a single-fold Diophantine representation are dominated by a limit-computable function $f:\mathbb{N}\setminus\{0\}\to\mathbb{N}$ which is implemented in MuPAD and whose computability is an open problem*, in: Computation, cryptography, and network security (eds. N. J. Daras, M. Th. Rassias), Springer, Cham, 2015, 577–590, *https://doi.org/10.1007/978-3-319-18275-9_24*.