On the Complexity of Conversion Between Classic Real Number Representations

Lars Kristiansen[1,2] and Jakob Grue Simonsen[3(✉)]

[1] Department of Mathematics, University of Oslo, Oslo, Norway
[2] Department of Informatics, University of Oslo, Oslo, Norway
[3] Department of Computer Science, DIKU, University of Copenhagen, Copenhagen, Denmark
simonsen@diku.dk

Abstract. It is known that while it is possible to convert between many different representations of irrational numbers (e.g., between Dedekind cuts and Cauchy sequences), it is in general not possible to do so subrecursively: conversions in general need to perform unbounded search. This raises the question of categorizing the pairs of representations between which either subrecursive conversion is possible, or is not possible.

The purpose of this paper is to prove the following positive result: for a number of well-known representations (Beatty sequences, Dedekind cuts, General base expansions, Hurwitz characteristics, and Locators) conversion between the representations can be performed effectively and with good subrecursive bounds.

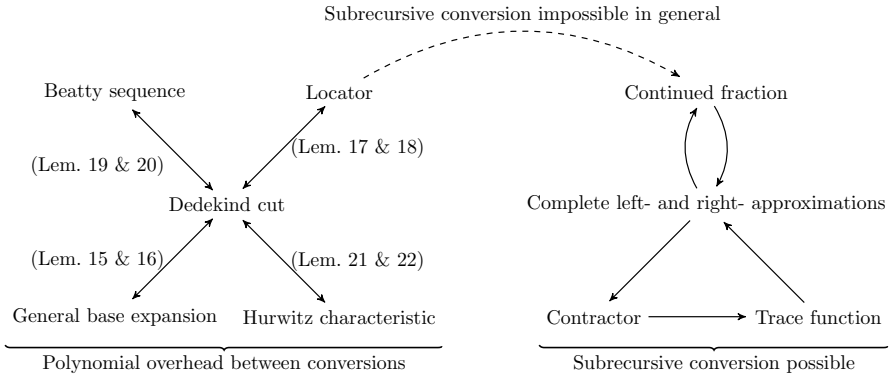
Keywords: Computable analysis · Computational complexity · Subrecursion · Representation of irrational numbers

The benefits of various representations of real numbers by computable functions is well-studied [10,16–18,21–23], and it is a standard result that the set of "computable reals" is the same in most representations, but that uniformly computable conversion between different representations is not always possible [18,22]. When computable conversion *is* possible, it is in general necessary to perform unbounded search, and efficient, or *subrecursive*, conversion cannot be done. For example, for any sufficiently large subrecursive class $\mathcal{S}$ of functions satisfying mild conditions (e.g., the set of primitive recursive functions or the set of Kalmár elementary functions), write $\mathcal{S}_F, \mathcal{S}_D$, and $\mathcal{S}_C$ for the sets of irrational numbers representable by continued fractions, Dedekind cuts, and rapidly converging Cauchy sequences computable by functions in $\mathcal{S}$. Then it is known that $\mathcal{S}_F \subsetneq \mathcal{S}_D \subsetneq \mathcal{S}_C$ [13,21], and thus *a fortiori* there can in general be no $\mathcal{S}$-computable uniform conversion from the Cauchy representation to the Dedekind cut representation, or from the Dedekind cut representation to the continued fraction representation (for, if $\mathcal{S}$ is closed under composition, such uniform conversions would imply $\mathcal{S}_F = \mathcal{S}_D = \mathcal{S}_C$).

M. Anselmo et al. (Eds.): CiE 2020, LNCS 12098, pp. 75–86, 2020.
https://doi.org/10.1007/978-3-030-51466-2_7

In this paper we derive upper bounds on the computational complexity of conversion between various representations of irrational numbers where subrecursive conversion *is* possible. In general, an irrational α in some representation R_1 will be computable by some function f (e.g., for the continued fraction representation $[3; 7, 15, 1, 292, \ldots]$ of π, $f(0) = 3, f(1) = 7, \ldots$), and in some other representation R_2 by some function g (e.g., for the decimal expansion of π, $g(0) = 3, g(1) = 1, g(2) = 4$); we are interested in the computational resources required to compute $g(n)$ when given access to the function f—in general this will require both querying the function f a number of times and performing a number of other operations, which we collectively call the "overhead" of conversion. In addition to the intrinsic value of this is the consequence that, roughly, if the overhead is a function in $\mathcal{S}$, and $\mathcal{S}$ satisfies natural closure properties, it follows that $\mathcal{S}_{R_1} \subseteq \mathcal{S}_{R_2}$.

The results of the paper are shown in the diagram below; all results in the *left-hand side* of the diagram are proved explicitly in the paper. The arrows in the *right-hand side* of the diagram are known from the literature [13–15]; we defer more precise bounds on these (to wit, the existence of primitive recursive bounds) to future research.



For the purposes of the present paper, we are only interested in *upper bounds* on conversion overhead. Our results show that conversion between Dedekind cuts and other classic representations can be done with polynomial overhead (and, by composition, conversion between any two of the considered representations in the left-hand side of the diagram above can be done with exponential overhead). More involved algorithms than the one we present here can indubitably be made, forcing the upper bounds to be low-degree polynomials.

Certain prior results are known for the representations we consider, but at a much coarser level of granularity; for example, Lehman [17] proved that the Hurwitz characteristic of α is primitive recursive iff the Dedekind cut of α is primitive recursive.

Remark 1. The overhead we consider is polynomial in the *value* of the index of the desired approximation to an irrational; for example, if one wants to compute

the nth digit in the base-b expansion of an irrational $\alpha \in (0,1)$ from a Dedekind cut of α, the overhead will be polynomial in n and b (as opposed to polynomial in the binary representations of n and b). Note that accordingly, the conversions we consider are thus computable in exponential time in the binary representations of n and b.

Remark 2. As the representations in the above diagrams are most easily expressed using functions, we believe that the natural formalizations for conversions are Turing machines with oracle access to the representations being converted from. In other work on real number computation, there is a well-developed notion of reducibility between representations that, roughly, requires the representation to be written as an infinite string on one of the input tapes of a type-2 Turing machine [6,12,22,24]. In that setting, for example, a function $f : \mathbb{Q} \cap [0,1] \longrightarrow \{0,\ldots,b-1\}$ is most naturally expressed by imposing a computable ordering on its domain (e.g., rationals appear in non-decreasing order of their denominator), and the function values $f(q)$ appear encoded as bit strings in this order. We strongly conjecture that our results carry over to the type-2 setting *mutatis mutandis*.

1 Preliminaries

We assume basic familiary with computability and computational complexity (standard textbooks are [1,8,20]). We write $f(n) = \mathrm{poly}(n)$ if $f : \mathbb{N} \longrightarrow \mathbb{N}$ is bounded above by a polynomial in n with positive integer coefficients, and $f(n) = \mathrm{polylog}(n)$ if f is bounded above by a polynomial in $\log n$ with positive integer coefficients.

We first define oracle machines in the usual way:

Definition 3. *A (parameterized)* function-oracle Turing machine *is a (multi-tape) Turing machine $M = (Q, q_0, F, \Sigma, \Gamma, \delta)$ with initial state $q_0 \in Q$, final states $F \subseteq Q$, input and tape alphabets Σ and Γ (with $\Sigma \subseteq \Gamma$ and $\{\textvisiblespace\} \subseteq \Gamma \setminus \Sigma$), and partial transition function δ such that M has a special* query tape *and two distinct states $q_q, q_a \in Q$ (the* query *and* answer *states).*

To be executed, M is provided with a total function $f : \mathbb{N} \longrightarrow \mathbb{N}$ (the oracle) prior to execution on any input. We write M^f for M when f has been fixed–note that M^f is then a usual (non-parameterized) function-oracle machine [8]. The transition relation of M^f is defined as usual for Turing machines, except for the state q_q: If M enters state q_q, let m be (a representation in the tape alphabet of) the value currently on the query tape; M moves to state q_a in a single step, and the contents of the query tape are instantaneously changed to (the representation in the tape alphabet of) $f(m)$. The time- and space complexity *of a function-oracle machine is counted as for usual Turing machines, with the transition between q_q and q_a taking $|f(m)|$ time steps, and the space use of the query tape after the transition being $|f(m)|$. The* (input) size *of a query is the number of symbols on the query tape when M enters state q_q.*

Function-oracle machines are in standard use in complexity theory of functions on the set of real numbers (see, e.g., [11]).

Remark 4. All input and output tapes of Turing machines are assumed to have alphabet $\{0,1\}$ in addition to the blank symbol. All work tapes have alphabet $\{0,1,2\}$ in addition to the blank symbol. Unless otherwise stated, all elements of $\mathbb{N}$, $\mathbb{Z}$, and elements of any finite set, are assumed to be written on input, query, and output tapes in their binary representation. Pairs (p,q) of integers are assumed to be written using interleaved notation (i.e., the first bit of the binary representation of p followed by the first bit of the binary representation of q, and so forth). In case the representations have unequal length, the shortest binary representation is padded with '2' in the interleaving. Observe that the length of the representation of a pair (p,q) is then $O(\log \max\{p,q\})$. All elements $p/q \in \mathbb{Q}$ are assumed to be represented by the representation of (p,q).

As expected, using the semantic function of a function-oracle Turing machine M with oracle to f as the oracle of another function-oracle Turing machine N can be made to "cut out the middleman machine M"; that is, we could use a single oracle machine with an oracle to f with bounds on time and oracle use not much higher than the original machines M and N:

Proposition 5 (Compositionality). *Let M and N be parameterized function-oracle machines and let f be a map. Write $g = \phi_{M^f} : A \longrightarrow B$ and $\phi_{N^g} : C \longrightarrow D$ for sets A, B, C, D (all representable by elements of $\{0,1\}^+$). Let $t_M, t_N, q_M, q_N, s : \mathbb{N} \longrightarrow \mathbb{N}$ be maps. Suppose that*

1. *M^f on input $a \in A$ computes $g(a)$ in time $t_M(|a|)$ using $q_M(|a|)$ queries to f, and*
2. *N^g on input $c \in C$ computes ϕ_{N^g} in time $t_N(|c|)$ using $q_N(|c|)$ queries to g, each of input size at most $s(|c|)$.*

Then there is a parameterized function-oracle machine P such that $\phi_P^f = \phi_{N^g}$, and P^f on input $c \in C$ runs in time at most

$$O(q_N(|c|)t_M(s(|c|)) + t_N(|c|))$$

using at most

$$q_N(|c|)q_M(s(|c|))$$

queries to f.

Proof. P is merely N with the original oracle tape replaced by two new work tapes, a new oracle tape added, and each query to g replaced by execution of a copy of M, with the new work tapes functioning as the "input" and "output" tapes of the copy of M, and the new oracle tape as the oracle tape of the copy of M. Every time N would query g, it writes the query a on the new "input" work tape. The copy of M then computes $g(a)$ using time at most $t_M(s(|c|))$ with $q_M(s(|c|))$ queries to f, hence a total of $q_N(|c|)q_M(s(|c|))$ queries to f. The total time spent by P is the time spent by N plus at most $t_M(s(|c|))$ steps per oracle query, for a total of $O(q_N(|c|)t_M(s(|c|)) + t_N(|c|))$ steps. □

In particular, function-oracle machines running in polynomial time and having queries of polynomial input size are composable in the above way and yield new machines running in polynomial time.

1.1 Farey Sequences and the Stern-Brocot Tree

A *Farey sequence* is a strictly increasing sequence of fractions between 0 and 1. The Farey sequence of *order* k, denoted F_k, contains all fractions which when written in their lowest terms, have denominators less than or equal to k.

Let a/b and c/d be fractions in their lowest terms. The fraction $m(a/b, c/d) = (a+c)/(b+d)$ is called the *mediant* of a/b and c/d. The ordered pair of two consecutive fractions in a Farey sequence is called a *Farey pair*. It is easy to see that $a/b < m(a/b, c/d) < c/d$ if $a/b \neq c/d$.

We arrange the fractions strictly between 0 and 1 in a binary search tree $\mathcal{T}_{\mathrm{F}}$.

Definition 6. *The* Farey pair tree $\mathcal{T}_{\mathrm{F}}$ *is the complete infinite binary tree where each node has an associated Farey pair* $(a/b, c/d)$ *defined by recursion on the position* $\sigma \in \{0,1\}^*$ *of a node in* $\mathcal{T}_{\mathrm{F}}$ *as follows:* $\mathcal{T}_{\mathrm{F}}(\epsilon) = (0/1, 1/1)$*, and if* $\mathcal{T}_{\mathrm{F}}(\sigma) = (a/b, c/d)$*, then* $\mathcal{T}_{\mathrm{F}}(\sigma 0) = (a/b, (a+c)/(b+d))$ *and* $\mathcal{T}_{\mathrm{F}}(\sigma 1) = ((a+c)/(b+d), c/d)$*. The* depth *of a node in* $\mathcal{T}_{\mathrm{F}}$ *is the length of its position (with the depth of the root node being* 0*).*

Abusing notation slightly, we do not distinguish between the pair $\mathcal{T}_{\mathrm{F}}(\sigma) = (a/b, c/d)$ *and the open interval* $(a/b, c/d)$*.*

The (left) Stern-Brocot tree[1] $\mathcal{T}_{\mathrm{SB}}$ *is the infinite binary tree obtained by the Farey pair tree where each Farey pair* $(a/b, c/d)$ *has been replaced by its mediant* $m(a/b, c/d) = (a+c)/(b+d)$*.*

Thus, we have, for example:

$$\mathcal{T}_{\mathrm{F}}(0) = \left(\frac{0}{1}, \frac{1}{2}\right), \mathcal{T}_{\mathrm{F}}(1) = \left(\frac{1}{2}, \frac{1}{1}\right), \mathcal{T}_{\mathrm{F}}(10) = \left(\frac{1}{2}, \frac{2}{3}\right), \mathcal{T}_{\mathrm{F}}(0000) = \left(\frac{0}{1}, \frac{1}{5}\right)$$

We shall later need the two following ancillary propositions which we include without proof.

Proposition 7. *Let* $p/q \in \mathbb{Q} \cap [0,1]$ *be a fraction in its lowest terms. Then,* p/q *is a fraction in a Farey pair at depth at most* $p + q - 1$ *in* $\mathcal{T}_{\mathrm{F}}$*.*

Efficient computations of the elements of the Stern-Brocot tree (and hence also the Farey pair tree) is possible [2,19]; for our purposes, we simply need the following result:

Proposition 8. *There is a Turing machine* M *such that for any* $\sigma \in \{0,1\}^*$*,* $\phi_M(\sigma) = \mathcal{T}_{\mathrm{F}}(\sigma)$ *and* M *runs in time* $\mathrm{poly}(1 + |\sigma|)$*.*

[1] "Left" because the Stern-Brocot tree originally concerns the interval $(0, 2)$ and we are interested only in $(0, 1)$ which corresponds to the left child of the Stern-Brocot tree.

2 Representations

We now introduce a number of well-known representations of real numbers. Representations by Dedekind cuts [4,7], Beatty sequences [3][2], and Hurwitz characteristics [9][3] were known in the 19th century or earlier. The representations by locators and general base expansions are, to our knowledge, new, but natural. In particular, the general base expansion yields the base-b expansions of α in *any* integer base $b \geq 2$ on demand; it turns out that this is the key to subrecursive equivalence with Dedekind cuts (whereas the base-b expansion for any *fixed* b is not subrecursively equivalent to Dedekind cuts, see [14]).

Definition 9. *A function* $D : \mathbb{Q} \longrightarrow \{0,1\}$ *is the (left)* Dedekind cut *of the irrational number* α *when* $D(q) = 0$ *iff* $q < \alpha$.

Definition 10. *A function* $B : \mathbb{N} \longrightarrow \mathbb{Z}$ *is the (function computing the)* Beatty sequence *of the irrational number* α *when*

$$\frac{B(q)}{q} \quad < \quad \alpha \quad < \quad \frac{B(q)+1}{q}$$

Definition 11. *A function* $L_\alpha : \mathbb{Q} \times \mathbb{Q} \longrightarrow \{0,1\}$ *is the* locator *of the real number* α *when* $L_\alpha(p,q) = 0$ *iff* α *is in the interval* (p,q).

Definition 12. *Let* $(0.\mathtt{D}_1\mathtt{D}_2\ldots)_b$ *be the base-b expansion of the real number* α. *We define the function* $E_b^\alpha : \mathbb{N} \longrightarrow \{0,\ldots,b-1\}$ *by* $E_b^\alpha(0) = 0$ *and* $E_b^\alpha(i) = \mathtt{D}_i$ *(for* $i \geq 1$*).*

A general base expansion *of the real number* α *is the function*

$$E : (\mathbb{N} \setminus \{0,1\}) \times \mathbb{N} \longrightarrow \{0,\ldots,b-1\}$$

where $E(b,q) = E_b^\alpha(q)$.

Definition 13. *The* Hurwitz *characteristic of the irrational number* $\alpha \in (0,1)$ *is the map* $H : \mathbb{N} \longrightarrow \{0,1\}^*$ *such that* $H(0), H(1), H(2), \ldots$ *is a path in the Stern-Brocot three, and:* [4]

$$\alpha = \lim_{q\to\infty} m(\mathcal{T}_{\mathrm{F}}(H(q))) = \lim_{q\to\infty} \mathcal{T}_{\mathrm{SB}}(H(q))$$

[2] Apparently, what is now known as Beatty sequences was used earlier by Bernard Bolzano [5], whence this representation of reals could also be called *Bolzano measures*.

[3] Use of the Hurwitz characteristic to represent numbers rather than a stepping stone for other material is a much younger invention [17].

[4] Strictly speaking, the classic Hurwitz characteristic corresponds to a path through the full Stern-Brocot tree (not just the "left" tree as we consider here), and hence the classic Hurwitz characteristic H' of $\alpha \in (0,1)$ is the function defined by $H'(0) = 0$ and $H'(q) = 0 \cdot H(q-1)$ for $q > 0$. This does not change our results in any material way.

3 Representations Subrecursively Equivalent to Dedekind Cuts

The remainder of the paper is devoted to proving the following theorem:

Theorem 14. *For each representation R below there is a parameterized function-oracle machine M such that, for every irrational α between 0 and 1, M when provided with an oracle to the R-representation of α, will compute the Dedekind cut of R, and N when provided with an oracle to the Dedekind cut of α, will compute the R-representation of α. Let n be the largest integer in the input (i.e., n if domain of R is $\mathbb{N}$, $n = \max\{n_1, n_2\}$ if the domain of R is $\mathbb{N} \times \mathbb{N}$, $\max\{p, q\}$ if the domain of R is $\mathbb{Q}$, and $\max\{p_1, q_1, p_2, q_2\}$ if the domain of R is $\mathbb{Q} \times \mathbb{Q}$). Then, M and N will produce their output in time $\mathrm{poly}(n)$ using at most $\mathrm{poly}(n)$ oracle calls of size at most $\mathrm{poly}(n)$.*

- *the locator of α*
- *the Beatty sequence of α*
- *the general base expansion of α*
- *the Hurwitz characteristic of α*

Furthermore, conversion between any two of the above representations (e.g., from the locator of α to the Beatty sequence of α) can be done by function-oracle machines producing their output using exponential (in n) time, exponential (in n) number of oracle calls, and exponential (in n) size of oracle calls.

The proof of conversion from and to Dedekind cuts is contained in the sequence of lemmas below that all relate the various representations to the Dedekind cut. All lemmas assert existence of parameterized function-oracle machines that will convert *from* or *to* the Dedekind cut of α with polynomial overhead (often with smaller overhead, whence the result follows *a fortiori*). The result that we can convert between any of the representations using exponential overhead then follows by an application of Proposition 5.

3.1 Conversion Between General Base Expansions and Dedekind Cuts

Lemma 15. *There is a parameterized function-oracle Turing machine M such that if $D : \mathbb{Q} \longrightarrow \{0, 1\}$ is the Dedekind cut of any irrational number $\alpha \in (0, 1)$, then $\phi_M^D : (\mathbb{N} \setminus \{0, 1\}) \times \mathbb{N} \longrightarrow \{0, \ldots, b-1\}$ is the general base expansion of α. Moreover, on input $(b, n) \in (\mathbb{N} \setminus \{0, 1\}) \times \mathbb{N}$, M^D runs in time $O(b^2 \mathrm{poly}(n))$, and uses at most $n \log_2 b$ oracle calls, each of input size at most $O(n \log_2 b)$ bits.*

Proof. M constructs the sequence $E_b^\alpha(1), E_b^\alpha(2), \ldots, E_b^\alpha(n)$ inductively by maintaining an open interval $I_i = (v_i, w_i)$ with rational endpoints $v_i, w_i \in \mathbb{Q}$ for each $i \in \{0, \ldots, n-1\}$ such that (i) $\alpha \in I_i$, (ii) v_i is a multiple of b^{-i}, and (iii) $w_i - v_i = b^{-i}$. Initially, $I_0 = (0, 1)$. For each interval I_i, M splits $I_i = (v_i, w_i)$ into b equal-sized intervals

$$(v_i, v_i + b^{-(i+1)}), \ldots, (v_i + (b-1)b^{-(i+1)}, v_i + b^{-i}) = (v_i + (b-1)b^{-(i+1)}, w_i)$$

Observe that, for any interval (r_1, r_2), if $D(r_1) = D(r_2) = 0$, then $\alpha > r_2$, and if $D(r_1) = D(r_2) = 1$, then $\alpha < r_1$ (and the case $D(r_1) = 1 \wedge D(r_2) = 0$ is not possible). Thus, M can use D to perform binary search on (the endpoints of) the above set of intervals to find the (necessarily unique) interval $(v_i + jb^{-(i+1)}, v_i + (j+1)b^{-(i+1)})$ that contains α (observe that, for this interval, $D(v_i+jb^{-(i+1)}) = 0$ and $D(v_i+(j+1)b^{-(i+1)}) = 1$). We then set $(v_{i+1}, w_{i+1}) = (v_i+jb^{-(i+1)}, v_i+(j+1)b^{-(i+1)})$. By construction, we have $E^\alpha(b, i+1) = j$. Clearly, in each step i, there are at most $\log_2 b$ oracle calls to D, and the construction of each of the b intervals and writing on the query tape can be performed in time polynomial in the binary representation of the numbers involved, hence in time $O(\text{polylog}(b^i)) = O(\text{poly}(i)\text{polylog}(b))$. Hence, the total time needed to produce $E(b, n)$ is at most $O(bn\text{poly}(n)\text{polylog}(b)) = O(b^2\text{poly}(n))$ with at most $n \log_2 b$ queries to D. In each oracle call, the rational numbers involved are all endpoints of intervals where the endpoints are sums of negative powers of b and where the exponent of all powers are at most n. Hence, all oracle calls can be represented by rational numbers using at most $O(n \log_2 b)$ bits. □

Lemma 16. *There is a parameterized function-oracle Turing machine M such that if $E : (\mathbb{N} \setminus \{0, 1\}) \times \mathbb{N} \longrightarrow \{0, \ldots, b-1\}$ is the general base expansion of α, then $\phi^E_M : \mathbb{Q} \longrightarrow \{0, 1\}$ is the Dedekind cut of α. Moreover, on input $p/q \in \mathbb{Q}$, M^E runs in time $O(\log(\max\{p, q\}))$, and uses exactly 1 oracle call of input size at most $O(\log(q))$.*

Proof. On input $p/q \in \mathbb{Q}$, M first checks if $q = 1$, and outputs 0 if $p \leq 0$ and 1 if $p \geq 1$. Otherwise, $q > 1$, and M computes $E(q, 1)$; by definition, this is an element of $\{0, \ldots, q-1\}$. Observe that $D(p/q) = 0$ iff $p/q < \alpha$ iff $p \leq E(q, 1)$. Hence, M outputs 0 if $p \leq E(q, 1)$, and outputs 1 otherwise. M needs to write the (representation of the) pair $(q, 1)$ on the oracle tape and perform a single comparison of numbers of magnitude at most $\max\{p, q\}$, hence M uses time $O(\log \max\{p, q\})$ for the comparison. M uses exactly one oracle call to E with the pair $(q, 1)$, the representation of which uses at most $O(\log q)$ bits. □

3.2 Conversion Between Locators and Dedekind Cuts

Lemma 17. *There is a parameterized function-oracle Turing machine M such that if $D : \mathbb{Q} \longrightarrow \{0, 1\}$ is the Dedekind cut of any irrational number $\alpha \in (0, 1)$, then $\phi^D_M : \mathbb{Q} \times \mathbb{Q} \longrightarrow \{0, 1\}$ is the locator of α. Moreover, on input $(p_1/q_1, p_2/q_2) \in \mathbb{Q} \times \mathbb{Q}$, M^D runs in time $O(\log(\max\{p_1, q_1, p_2, q_2\}))$, and uses at most 2 oracle calls, each of input size at most $O(\log \max\{p_1, q_1, p_2, q_2\})$.*

Proof. Let L be the locator of α. Observe that for any two rational numbers $p_1/q_1, p_2/q_2 \in \mathbb{Q}$, we have $\alpha \in (p_1/q_1, p_2/q_2)$ iff $L(p_1/q_1, p_2/q_2) = 0$ iff $(D(p_1/q_1) = 0 \wedge D(p_2/q_2) = 1)$. Hence, M simply queries D (using the binary representations of the rationals) twice, outputs 1 if $(D(p_1/q_1) = 0 \wedge D(p_2/q_2) = 1)$, and outputs 1 otherwise. Clearly, the time needed is the time needed to transfer p_1/q_1 and p_2/q_2 to the query tape plus some constant independent of the size of the input, hence M uses time $O(\log(\max\{p_1, q_1, p_2, q_2\}))$. □

Lemma 18. *There is a parameterized function-oracle Turing machine M such that if $L : \mathbb{Q} \times \mathbb{Q} \longrightarrow \{0,1\}$ is the locator of any irrational number $\alpha \in (0,1)$, then $\phi_M^L : \mathbb{Q} \longrightarrow \{0,1\}$ is the Dedekind cut of α. Moreover, on input $p/q \in \mathbb{Q}$, M^L runs in time $O(\text{polylog}(\max\{p,q\}))$, and uses at most 1 oracle call of input size at most $O(\log \max\{p,q\})$.*

Proof. Observe that for $p/q \in (0,1) \cap \mathbb{Q}$, we have $D(p/q) = L(p/q, 1)$. Hence, M may, on input p/q simply perform a single query to L; this requires copying its input to the oracle tape, i.e. only linear time in the size of the representation of the input. By convention, the input p/q is representable in at most $O(\log \max\{p,q\})$ bits. □

3.3 Conversion Between Beatty Sequences and Dedekind Cuts

Lemma 19. *There is a parameterized function-oracle Turing machine M such that if $D : \mathbb{Q} \longrightarrow \{0,1\}$ is the Dedekind cut of any irrational number $\alpha \in (0,1)$, then $\phi_M^D : \mathbb{N} \longrightarrow \mathbb{Z}$ is the Beatty sequence of α. Moreover, on input $n \in \mathbb{N}$, M^D runs in time $O(\text{polylog}(n))$, and uses at most $\lceil \log n \rceil$ oracle calls to D, each of input size at most $O(\log n)$.*

Proof. On input n, M finds the least $i \in \{1, \ldots, n\}$ such that $D(\frac{i}{n}) = 1$. As $D(\frac{i}{n}) = 1$ and $j > i$ implies $D(\frac{j}{n}) = 1$, the least i can be found by binary search, halving the search range in each step[5]. This can be done by maintaining two integers l and u ranging in $\{1, \ldots, n\}$, and requires a maximum of $\log n$ halving steps. In each halving step, M finds the midpoint m between l and u, writes its binary representation on the query tape, queries D, and records the answer. Then, l and u are updated using basic binary arithmetic operations on integers, represented by at most $O(\log n)$ bits—if $D(m/n) = 1$, $u := m$, and if $D(m/n) = 0$, $l := m$. Clearly, in each step, the arithmetic and update operations can be performed in time polynomial in the size of the representation of the integers, hence in time polylog(n). As $(i-1)/n < \alpha < i/n$, we have $B(n) = i - 1$, and M^D thus returns $i - 1$. □

Lemma 20. *There is a parameterized function-oracle Turing machine M such that if $B : \mathbb{N} \longrightarrow \mathbb{Z}$ is the Beatty sequence of any irrational number $\alpha \in (0,1)$, then $\phi_M^B : \mathbb{Q} \longrightarrow \{0,1\}$ is the Dedekind cut of α. Moreover, on input $p/q \in \mathbb{Q}$, M^B runs in time $O(\log(\max\{p,q\}))$, and uses exactly one oracle call of input size $O(\log q)$.*

Proof. Observe that the Dedekind cut D of α satisfies $D(p/q) = 0$ if $p \leq B(q)$, and $D(p/q) = 1$ if $p > B(q)$. Thus, M may perform the oracle call $B(q)$ just once, resulting in an integer $B(q)$ (where $B(q) \in \{0, 1 \ldots, q-1\}$). The comparison $p \leq B(q)$ can be performed bitwise using the binary representations of p and $B(q)$ which is clearly linear in $\log(\max\{B(q), p\}) \leq \log \max\{p,q\}$. Writing q on the oracle tape clearly also takes time linear in $\log q$. □

[5] Observe that a brute-force search is also possible, yielding at most n oracle calls with input size at most $O(\log n)$ and obviating the need to reason about arithmetic operations.

3.4 Conversion Between Hurwitz Characteristics and Dedekind Cuts

Lemma 21. *There is a parameterized function-oracle Turing machine M such that if $H : \mathbb{N} \longrightarrow \{0,1\}^*$ is the Hurwitz characteristic of any irrational number $\alpha \in (0,1)$, then $\phi_M^H : \mathbb{Q} \longrightarrow \{0,1\}$ is the Dedekind cut of α. Moreover, on input $p/q \in \mathbb{Q}$, M^H runs in time* $\mathrm{poly}(\max\{p,q\})$*, and uses exactly one oracle call of input size at most $O(\log \max\{p,q\})$.*

Proof. On input $p/q \in \mathbb{Q}$ (where we assume wlog. that p/q is reduced to lowest terms), M computes $H(p+q)$ (using $\mathrm{polylog}(\max\{p,q\})$ operations to compute the binary representation of $p+q$, and then performing a single oracle call; note that the result of the oracle $H(p+q)$ is a bit string of length exactly $p+q = \mathrm{poly}(\max\{p,q\})$. M then computes $\mathcal{T}_{\mathrm{F}}(H(p+q))$ (by Proposition 8 this can be done in time $\mathrm{poly}(1+|H(p+q)|) = \mathrm{poly}(\max\{p,q\})$) to obtain a Farey pair $(a/b, c/d)$ such that $a/b < \alpha < c/d$. By Proposition 7, any reduced fraction p/q occurs as one of the fractions in a Farey pair in $\mathcal{T}_{\mathrm{F}}$ at depth at most $p+q-1$, and thus exactly one of (i) $p/q \leq a/b$ and (ii) $c/d \leq p/q$ must hold. Observe that $D(p/q) = 0$ iff $p/q \leq a/b$. Whether (i) or (ii) holds can be tested in time $O(\log \max\{a,b,c,d,p,q\})$. It is an easy induction on the depth d to see that a numerator or denominator in any fraction occurring in a Farey pair at depth d in $\mathcal{T}_{\mathrm{F}}$ is at most 2^d. Hence, $\max\{a,b,c,d,p,q\} \leq 2^{p+q}$, and the test can thus be performed in time $O(p+q) = O(\max\{p,q\})$. Thus, M needs a total time of $\mathrm{poly}(\max\{p,q\})$. □

Lemma 22. *There is a parameterized function-oracle Turing machine M such that if $D : \mathbb{Q} \longrightarrow \{0,1\}$ is the Dedekind cut of any irrational number $\alpha \in (0,1)$, then $\phi_M^D : \mathbb{N} \longrightarrow \{0,1\}^*$ is the Hurwitz characteristic of α. Moreover, on input $n \in \mathbb{N}$, M^D runs in time* $\mathrm{poly}(n)$*, and uses exactly n oracle calls, each of input size at most $O(n)$.*

Proof. On input n, M constructs a path of length n in the tree $\mathcal{T}_{\mathrm{SB}}$ corresponding to the bit string $H(n)$ by building the corresponding path in $\mathcal{T}_{\mathrm{F}}$. M can do this by starting at $i = 0$ and incrementing i, maintaining a *current* Farey pair $(a_i/b_i, c_i/d_i)$ such that $\alpha \in (a_i/b_i; c_i/d_i)$ for $i = 0, \ldots, n$ as the mediant of $(a_i/b_i, c_i/d_i)$ gives rise to the two children $p_L = (a_i/b_i, (a_i+c_i)/(b_i+d_i))$ and $p_R = ((a_i+c_i)/(b_i+d_i), c_i/d_i)$ of $(a_i/b_i, c_i/d_i)$ in $\mathcal{T}_{\mathrm{F}}$. Because α is irrational, it must be in exactly one of the open intervals $(a_i/b_i, (a_i+c_i)/(b_i+d_i))$ and $((a_i+c_i)/(b_i+d_i), c_i/d_i)$, and thus $(a_{i+1}/b_{i+1}, c_{i+1}/d_{i+1})$ must be either p_L or p_R. Clearly, $\alpha \in (a_i/b_i, (a_i+c_i)/(b_i+d_i))$ iff $D(a_i+c_i/b_i+d_i) = 1$ iff the ith bit of $H(n)$ is 0. Hence, M starts with $(a_0/b_0, c_0/d_0) = (0/1, 1/1)$, and constructs the n intervals $(a_i/b_i, c_i/d_i)$ for $i = 1, \ldots, n$ by computing the mediant and querying D in each step. Observe that the query in step i is the (binary representation of the) mediant of a Farey pair at depth $i-1$, thus its denominator is bounded above by 2^i and its binary representation uses at most $O(\log 2^i) = O(i)$ bits.

As the numerators and denominators at depth i in $\mathcal{T}_{\mathrm{F}}$ are of size at most 2^i (hence representable by i bits), computing the mediant at step i can be done

in time at most $O(i) = O(n)$ by two standard schoolbook additions, and the step i contains exactly one query to D. Hence, the total time needed for M to construct $H(n)$ is at most $O(n\text{poly}(n)) = \text{poly}(n)$, with exactly n oracle calls, each of size at most $O(\log 2^n) = O(n)$. □

4 Conclusion and Future Work

We have analyzed conversions between representations equivalent to Dedekind cuts, and we have seen that we can convert efficiently between any two such representations (Theorem 14) . We strongly conjecture that the same efficiency is not possible between representations equivalent to continued fractions. Indeed, we regard the representations equivalent to continued fractions to be the most interesting and challenging ones from a mathematical point of view. Among these representations we find the trace functions and the contractors (see the figure on Page 2). A function $T : \mathbb{Q} \to \mathbb{Q}$ is a *trace function* for the irrational number α when $|\alpha - r| > |\alpha - T(r)|$. A function $F : [0, 1] \to (0, 1)$ is a *contractor* if we have $|F(r_1) - F(r_2)| < |r_1 - r_2|$ for any rationals r_1, r_2 where $r_1 \neq r_2$. Both trace functions and contractors can be converted to (and from) continued fractions without unbounded search, and converting from a contractor to a trace function is easy as it can be proved that every contractor is a trace function. But conversely, we believe that it is not possible to convert a trace function to a contractor within reasonably small time or space bounds.

Conversions between representations equivalent to rapidly converging Cauchy sequences also deserve a further study. One such representation will be base-2 expansions over the digits 0 (zero), 1 (one) and $\overline{1}$ (minus one). In this representation, the rational number $1/4$ can be written as 0.01, but also as $0.1\overline{1}$. Another interesting representation are the *fuzzy Dedekind cuts*. A fuzzy Dedekind cut for an irrational number α is a function $D : \mathbb{Z} \times \mathbb{N} \to \{0, 1\}$ satisfying (i) $D(p, q) = 0$ implies $\alpha < (p + 1)/q$ and (ii) $D(p, q) = 1$ implies $(p - 1)/q < \alpha$. Thus, each irrational α will have (infinitely) many fuzzy Dedekind cuts. If D is a fuzzy cut for α and we know that $D(3, 8) = 0$, then we know that α lies below $4/8$ (but we do not know if α lies below $3/8$). Moreover, if we also know that $D(6, 16) = 1$, then we know that α lies in the interval $(3/8 - 1/16, 3/8 + 1/8)$ (but we do not know if α lies below or above $3/8$).

Finally, some well-known representations are not subrecursively equivalent to any of the three representations above, for example the base-b representation for any integer base $b \geq 2$. It is possible to convert a Dedekind cut to a base-b expansion and a base-b expansion into a Cauchy sequence without unbounded search, but not the other way around [14,21]. It is interesting to investigate the set of representations subrecursively equivalent to such expansions.

Acknowledgment. We are grateful for the meticulous comments of one of the referees; these have helped to significantly improve the paper.

References

1. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press, Cambridge (2009)
2. Bates, B., Bunder, M., Tognetti, K.: Locating terms in the Stern-Brocot tree. Eur. J. Comb. **31**(3), 1020–1033 (2010)
3. Beatty, S., et al.: Problems for solutions: 3173–3180. Am. Math. Mon. **33**(3), 159–159 (1926)
4. Bertrand, J.: Traité d'arithmétique (1849)
5. Bolzano, B.: Pure Theory of Numbers. Oxford University Press, Oxford (2004). In the Mathematical Works of Bernard Bolzano edited and translated by Steve Russ, pp. 355–428
6. Brattka, V., Hertling, P.: Topological properties of real number representations. Theoret. Comput. Sci. **284**(2), 241–257 (2002)
7. Dedekind, R.: Stetigkeit und irrationale Zahlen. Vieweg, Braunschweig (1872)
8. Du, D.Z., Ko, K.I.: Theory of Computational Complexity. Wiley Interscience (2000)
9. Hurwitz, A.: Ueber die angenäherte Darstellung der Irrationalzahlen durch rationale Brüche. Mathematische Annalen **39**, 279–284 (1891)
10. Ko, K.: On the definitions of some complexity classes of real numbers. Math. Syst. Theory **16**, 95–109 (1983)
11. Ko, K., Friedman, H.: Computational complexity of real functions. Theoret. Comput. Sci. **20**(3), 323–352 (1982)
12. Kreitz, C., Weihrauch, K.: Theory of representations. Theor. Comput. Sci. **38**, 35–53 (1985)
13. Kristiansen, L.: On subrecursive representability of irrational numbers. Computability **6**, 249–276 (2017)
14. Kristiansen, L.: On subrecursive representability of irrational numbers, part ii. Computability **8**, 43–65 (2019)
15. Kristiansen, L.: On subrecursive representability of irrational numbers: continued fractions and contraction maps. In: Proceedings of the 16th International Conference on Computability and Complexity in Analysis (CCA 2019) (2019)
16. Labhalla, S., Lombardi, H.: Real numbers, continued fractions and complexity classes. Ann. Pure Appl. Logic **50**(1), 1–28 (1990)
17. Lehman, R.S.: On primitive recursive real numbers. Fundamenta Mathematica **49**(2), 105–118 (1961)
18. Mostowski, A.: On computable sequences. Fundamenta Mathematica **44**, 37–51 (1957)
19. Niqui, M.: Exact arithmetic on the Stern-Brocot tree. J. Discrete Algorithms **5**(2), 356–379 (2007)
20. Sipser, M.: Introduction to the Theory of Computation. PWS Publishing Company (1997)
21. Specker, E.: Nicht konstruktiv beweisbare Sätze der Analysis. J. Symb. Logic **14**(3), 145–158 (1949)
22. Weihrauch, K.: The degrees of discontinuity of some translators between representations of real numbers. Technical report, Fernuniversität Hagen (1992)
23. Weihrauch, K.: Computable Analysis. Springer, Heidelberg (2000). https://doi.org/10.1007/978-3-642-56999-9
24. Weihrauch, K., Kreitz, C.: Representations of the real numbers and of the open subsets of the set of real numbers. Ann. Pure Appl. Logic **35**, 247–260 (1987)