Dedekind Sums

Manuel Eberl, Anthony Bordg, Lawrence C. Paulson, Wenda Li

October 13, 2025

Abstract

For integers h, k, the Dedekind sum is defined as

$$s(h,k) = \sum_{r=1}^{k-1} \frac{r}{k} \left(\left\{ \frac{hr}{k} \right\} - \frac{1}{2} \right)$$

where $\{x\} = x - \lfloor x \rfloor$ denotes the fractional part of x.

These sums occur in various contexts in analytic number theory, e.g. in the functional equation of the Dedekind η function or in the study of modular forms.

We give the definition of $s(h,k)$ and prove its basic properties, including the reciprocity law

$$s(h,k) + s(k,h) = \frac{1}{12hk} + \frac{h}{12k} + \frac{k}{12h} - \frac{1}{4}$$

and various congruence results.

Our formalisation follows Chapter 3 of Apostol's *Modular Functions and Dirichlet Series in Number Theory* [1] and contains all facts related to Dedekind sums from it (without the exercises).

Contents

1 Dedekind sums **2**
1.1 Preliminaries 2
1.2 Definition and basic properties 2
1.3 The Reciprocity Law 4
1.4 Congruence Properties 5

1 Dedekind sums

```
theory Dedekind_Sums
imports
  Complex_Main
  "HOL-Library.Periodic_Fun"
  "HOL-Library.Real_Mod"
  "HOL-Number_Theory.Number_Theory"
  "Bernoulli.Bernoulli_FPS"
begin
```

1.1 Preliminaries

```
lemma rcong_of_intI: "[a = b] (mod m) ⟹ [of_int a = of_int b] (rmod
(of_int m))"
  ⟨proof⟩

lemma rcong_of_int_iff: "[of_int a = of_int b] (rmod (of_int m)) ⟷
[a = b] (mod m)"
⟨proof⟩
```

1.2 Definition and basic properties

```
definition dedekind_sum :: "int ⇒ int ⇒ real" where
  "dedekind_sum h k ≡ ∑r∈{1..<k}. (of_int r / of_int k * (frac (of_int
(h*r) / of_int k) - 1/2))"

definition dedekind_frac :: "real ⇒ real" where
  "dedekind_frac x = (if x ∈ ℤ then 0 else frac x - 1 / 2)"

lemma dedekind_frac_int [simp]: "x ∈ ℤ ⟹ dedekind_frac x = 0"
  ⟨proof⟩

notation dedekind_frac ("⟨_⟩")

interpretation dedekind_frac: periodic_fun_simple' dedekind_frac
⟨proof⟩

lemma dedekind_frac_uminus [simp]: "⟨-x⟩ = -⟨x⟩"
  ⟨proof⟩

lemma dedekind_frac_one_minus [simp]: "⟨1 - x⟩ = -⟨x⟩"
  ⟨proof⟩

lemma dedekind_frac_rcong:
  assumes "[x = x'] (rmod 1)"
  shows   "⟨x⟩ = ⟨x'⟩"
⟨proof⟩

lemma dedekind_frac_mod:
```

```
  "⟨of_int (a mod k) / of_int k⟩ = ⟨of_int a / of_int k⟩"
⟨proof⟩

lemma sum_dedekind_frac_eq_0:
  "(∑r∈{1..<k}. ⟨of_int r / of_int k⟩) = 0"
⟨proof⟩

lemma sum_dedekind_aux:
  assumes "f (0::int) = 0"
  shows   "(∑r∈{0..<k}. f r) = (∑r∈{1..<k}. f r)"
⟨proof⟩

lemma sum_dedekind_frac_eq_0':
  "(∑r∈{0..<k}. ⟨of_int r / of_int k⟩) = 0"
⟨proof⟩

lemma sum_dedekind_frac_mult_eq_0:
  assumes "coprime h k"
  shows   "(∑r∈{1..<k}. ⟨of_int (h * r) / of_int k⟩) = 0"
⟨proof⟩

lemma sum_dedekind_frac_mult_eq_0':
  assumes "coprime h k"
  shows   "(∑r∈{0..<k}. ⟨of_int (h * r) / of_int k⟩) = 0"
⟨proof⟩

lemma dedekind_sum_altdef:
  assumes "coprime h k"
  shows   "dedekind_sum h k = (∑r∈{1..<k}. ⟨of_int r / of_int k⟩ * ⟨of_int
(h*r) / of_int k⟩)"
⟨proof⟩


theorem dedekind_sum_cong:
  assumes "[h' = h] (mod k)"
  assumes "coprime h' k ∨ coprime h k"
  shows "dedekind_sum h' k = dedekind_sum h k"
⟨proof⟩

theorem dedekind_sum_negate:
  assumes "coprime h k"
  shows   "dedekind_sum (-h) k = -dedekind_sum h k"
⟨proof⟩

theorem dedekind_sum_negate_cong:
  assumes "[h' = -h] (mod k)" "coprime h' k ∨ coprime h k"
  shows "dedekind_sum h' k = -dedekind_sum h k"
⟨proof⟩
```

theorem *dedekind_sum_inverse:*
  assumes *"[h * h' = 1] (mod k)"*
  shows *"dedekind_sum h k = dedekind_sum h' k"*
⟨*proof*⟩

theorem *dedekind_sum_inverse':*
  assumes *"[h * h' = -1] (mod k)"*
  shows *"dedekind_sum h k = -dedekind_sum h' k"*
⟨*proof*⟩

theorem *dedekind_sum_eq_zero:*
  assumes *"[h^2 + 1 = 0] (mod k)"*
  shows *"dedekind_sum h k = 0"*
⟨*proof*⟩

1.3 The Reciprocity Law

theorem *sum_of_powers':*
  "(∑k<n::nat. (real k) ^ m) = (bernpoly (Suc m) n - bernpoly (Suc m) 0) / (m + 1)"
⟨*proof*⟩

theorem *sum_of_powers'_int:*
  assumes *"n ≥ 0"*
  shows *"(∑k=0..<n::int. real_of_int k ^ m) = (bernpoly (Suc m) n - bernpoly (Suc m) 0) / (m + 1)"*
⟨*proof*⟩

theorem *sum_of_powers'_int_from_1:*
  assumes *"n ≥ 0" "m > 0"*
  shows *"(∑k=1..<n::int. real_of_int k ^ m) = (bernpoly (Suc m) n - bernpoly (Suc m) 0) / (m + 1)"*
⟨*proof*⟩

theorem *dedekind_sum_reciprocity:*
  assumes *"h > 0"* **and** *"k > 0"* **and** *"coprime h k"*
  shows *"12 * h * k * dedekind_sum h k + 12 * k * h * dedekind_sum k h =
          h^2 + k^2 - 3 * h * k + 1"*
⟨*proof*⟩

theorem *dedekind_sum_reciprocity':*
  assumes *"h > 0"* **and** *"k > 0"* **and** *"coprime h k"*
  shows *"dedekind_sum h k = -dedekind_sum k h + h / k / 12 + k / h / 12 - 1 / 4 + 1 / (12 * h * k)"*
  ⟨*proof*⟩

1.4 Congruence Properties

definition *dedekind_sum' :: "int ⇒ int ⇒ int"* **where**
*"dedekind_sum' h k = ⌊6 * real_of_int k * dedekind_sum h k⌋"*

lemma *dedekind_sum'_cong:*
"[h = h'] (mod k) ⟹ coprime h k ∨ coprime h' k ⟹ dedekind_sum' h k = dedekind_sum' h' k"
⟨*proof*⟩

lemma
assumes *"k > 0"*
shows *of_int_dedekind_sum':*
*"real_of_int (dedekind_sum' h k) = 6 * real_of_int k * dedekind_sum h k"*
and *dedekind_sum'_altdef:*
*"dedekind_sum' h k = h * (k - 1) * (2 * k - 1) -
6 * (∑r = 1..<k. r * ⌊of_int (h * r) / k⌋) - 3 * (k * (k - 1) div 2)"*
and *dedekind_sum'_cong_3: "[dedekind_sum' h k = h * (k - 1) * (2 * k - 1)] (mod 3)"*
⟨*proof*⟩

lemma *three_dvd_dedekind_sum'_iff_aux:*
fixes *h k :: int*
defines *"ϑ ≡ gcd 3 k"*
assumes *"k > 0" "coprime h k"*
shows *"3 dvd (2 * dedekind_sum' h k) ⟷ ¬3 dvd k"*
⟨*proof*⟩

lemma *dedekind_sum'_reciprocity:*
fixes *h k :: int*
assumes *"h > 0" "k > 0" "coprime h k"*
shows *"2 * h * dedekind_sum' h k = -2 * k * dedekind_sum' k h +* h^2 *+* k^2 *- 3 * h * k + 1"*
⟨*proof*⟩

lemma *cong_dedekind_sum'_1:*
fixes *h k :: int*
defines *"ϑ ≡ gcd 3 k"*
assumes *"h > 0" "coprime h k"*
shows *"[2 * k * dedekind_sum' k h = 0] (mod ϑ * k)"*
⟨*proof*⟩

lemma *cong_dedekind_sum'_2_aux:*
fixes *h k :: int*
defines *"ϑ ≡ gcd 3 k"*
assumes *"h > 0" "k > 0" "coprime h k"*

```
  shows "[2 * h * dedekind_sum' h k = h^2 + 1] (mod ϑ * k)"
⟨proof⟩

lemma dedekind_sum'_negate:
  assumes "k > 0" "coprime h k"
  shows   "dedekind_sum' (-h) k = -dedekind_sum' h k"
⟨proof⟩

lemma cong_dedekind_sum'_2:
  fixes h k :: int
  defines "ϑ ≡ gcd 3 k"
  assumes "k > 0" "coprime h k"
  shows "[2 * h * dedekind_sum' h k = h^2 + 1] (mod ϑ * k)"
⟨proof⟩



theorem dedekind_sum'_cong_8:
  assumes "k > 0" "coprime h k"
  shows "[2 * dedekind_sum' h k =
          (k-1)*(k+2) - 4*h*(k-1) + 4*(∑r∈{1..<(k+1) div 2}. ⌊2*h*r/k⌋)]
(mod 8)"
⟨proof⟩

theorem dedekind_sum'_cong_8_odd:
  assumes "k > 0" "coprime h k" "odd k"
  shows "[2 * dedekind_sum' h k =
          k - 1 + 4*(∑r∈{1..<(k+1) div 2}. ⌊2*h*r/k⌋)] (mod 8)"
⟨proof⟩



lemma dedekind_sum'_cong_power_of_two:
  fixes h k k1 :: int and n :: nat
  assumes "h > 0" "k1 > 0" "odd k1" "n > 0" "k = 2 ^ n * k1" "coprime
h k"
  shows   "[2 * h * dedekind_sum' h k =
              h^2 + k^2 + 1 + 5 * k - 4 * k * (∑v=1..<(h+1) div 2. ⌊of_int
(2 * k * v) / h⌋)]
           (mod 2 ^ (n + 3))"
⟨proof⟩

lemma dedekind_sum'_cong_power_of_two':
  fixes h k k1 :: int
  assumes "h > 0" "k > 0" "even k" "coprime h k"
  shows   "[2 * h * dedekind_sum' h k =
              h^2 + k^2 + 1 + 5 * k - 4 * k * (∑v=1..<(h+1) div 2. ⌊of_int
(2 * k * v) / h⌋)]
           (mod 2 ^ (multiplicity 2 k + 3))"
```

⟨proof⟩

```
lemma dedekind_sum_diff_even_int_aux:
  fixes a b c d :: int assumes det: "a * d - b * c = 1"
  fixes q c1 r δ' :: int and δ :: real
  assumes a: "a > 0"
  assumes q: "q ∈ {3, 5, 7, 13}" and "c1 > 0"
  assumes c: "c = q * c1"
  defines "r ≡ 24 div (q - 1)"
  defines "δ' ≡ (2 * dedekind_sum' a c - (a + d)) - (2 * q * dedekind_sum'
a c1 - (a + d) * q)"
  defines "δ ≡ (dedekind_sum a c - (a+d)/(12*c)) - (dedekind_sum a c1
- (a+d)/(12*c1))"
  shows   "of_int δ' = 12 * c * δ" and "24 * c dvd r * δ'"
```
⟨proof⟩

```
theorem dedekind_sum_diff_even_int:
  fixes a b c d :: int assumes det: "a * d - b * c = 1"
  fixes q c1 r :: int and δ' :: "int ⇒ int" and δ :: "int ⇒ real"
  assumes q: "q ∈ {3, 5, 7, 13}" and "c1 > 0"
  assumes c: "c = q * c1"
  defines "r ≡ 24 div (q - 1)"
  defines "δ' ≡ (λa. 2 * dedekind_sum' a c - (a + d) - (2 * q * dedekind_sum'
a c1 - (a + d) * q))"
  defines "δ ≡ (λa. dedekind_sum a c - (a+d)/(12*c) - (dedekind_sum a
c1 - (a+d)/(12*c1)))"
  shows   "of_int (δ' a) = 12 * c * δ a"
    and   "24 * c dvd r * δ' a"
    and   "real_of_int r * δ a / 2 ∈ ℤ"
```
⟨proof⟩

```
no_notation dedekind_frac ("⟨_⟩")

end
```

References

[1] T. M. Apostol. *Modular Functions and Dirichlet Series in Number Theory.* Graduate Texts in Mathematics. Springer, 1990.